Supplementary Material to Curriculum Dropout

Pietro Morerio[1], Jacopo Cavazza[1,2,3], Riccardo Volpi[1,3], René Vidal[2] and Vittorio Murino[1,4]

[1]Pattern Analysis & Computer Vision (PAVIS) – Istituto Italiano di Tecnologia – *Genova, 16163, Italy*
[2]Department of Biomedial Engineering – Johns Hopkins University – *Baltimore, MD 21218, USA*
[3]Electrical, Electronics and Telecommunication Engineering and Naval Architecture Department (DITEN) – Università degli Studi di Genova – *Genova, 16145, Italy*
[4]Computer Science Department – Università di Verona – *Verona, 37134, Italy*

{pietro.morerio,jacopo.cavazza,riccardo.volpi,vittorio.murino}@iit.it, rvidal@cis.jhu.edu

Contents

1. Theoretical proofs

In this Section, we provide all theoretical proofs for the statements reported in the paper. §1.1 draws additional connections with regularization theory, demonstrating that Curriculum Dropout implements an adaptive regularization scheme. Eventually, in §1.2, we formally prove that our dropout strategy is naturally interpretable within the curriculum learning paradigm [1], as we assert in section 4 of the paper.

1.1. Adaptive Regularization

In our work, we posit that our curriculum dropout can be interpreted as a smooth manner of imposing regularization. Precisely, the increase rate of neurons suppressions act as a progressive rule to simplify the overall model, as to prevent overfitting. In this Section, we establish connections between curriculum dropout and regularization theory [5] with adaptive schemes [8, 4, 2, 14, 3]. In order to do so, consider a supervised regression or classification problem where the data $\mathbf{x}_1, \dots, \mathbf{x}_N \in \mathbb{R}^d$ are paired with corresponding labels $y_1, \dots, y_N \in \mathbb{R}$. Assume a least square fitting

$$\min_{\mathbf{w}\in\mathbb{R}^d} \sum_{i=1}^{N} (y_i - \mathbf{w}^\top \mathbf{x}_i)^2 = \min_{\mathbf{w}\in\mathbb{R}^d} \|\mathbf{y} - \mathbf{X}\mathbf{w}\|_2^2 \tag{1}$$

to fit a linear model to explain the data, being $\mathbf{y} = [y_1, \dots, y_N]^\top$ and $\mathbf{X}$ the $N \times d$ matrix, whose i-th row $[X_{i1}, \dots, X_{id}] = \mathbf{x}_i^\top$. In the following pages, we will care of providing theoretical results which guarantee that dropout regularization on the un-regularized L_2 loss rewrites as a modification of the original loss, by adding a (data-dependent) regularizing term. More precisely, we apply the dropout formulation provided by [9, 15] to a classical least squares data fitting. As the following result show, once dropout is applied on an unregularized L^2 minimization, the latter problem rewrites as adding to the same loss a data-dependent regularization term which is modulated by $\theta(1-\theta)$.

Theorem 1 (Dropout - least squares)**.** *According to [9, 15], the dropout problem on the least squares fitting (1), rewrites*

$$\min_{\mathbf{w}\in\mathbb{R}^d} \mathbb{E}_{\mathbf{r}} \left[\sum_{i=1}^{N} (y_i - \mathbf{w}^\top (\mathbf{r} \odot \mathbf{x}_i))^2 \right] = \min_{\mathbf{w}\in\mathbb{R}^d} \mathbb{E}_{\mathbf{r}} \|\mathbf{y} - \mathbf{X}\mathrm{diag}(\mathbf{r})\mathbf{w}\|_2^2, \tag{2}$$

being $\mathbf{r} = [r_1, \dots, r_d]$ *and* $r_j \sim \mathrm{Bernoulli}(\theta)$ *i.i.d. It results*

$$\mathbb{E}_{\mathbf{r}} \|\mathbf{y} - \mathbf{X}\mathrm{diag}(\mathbf{r})\mathbf{w}\|_2^2 = \theta(1-\theta)\|\mathrm{diag}(\mathbf{X}^\top\mathbf{X})^{1/2}\mathbf{w}\|_2^2 + \|\mathbf{y} - \theta\mathbf{X}\mathbf{w}\|_2^2 \tag{3}$$

$$= \theta(1-\theta)\|\mathbf{w}\|^2_{\mathrm{diag}(\mathbf{X}^\top\mathbf{X})} + \|\mathbf{y} - \theta\mathbf{X}\mathbf{w}\|_2^2 \tag{4}$$

Proof. Using the definition of the Euclidean norm and the linearity of the expected value, we get

$$\mathbb{E}_{\mathbf{r}} \|\mathbf{y} - \mathbf{X}\mathrm{diag}(\mathbf{r})\mathbf{w}\|_2^2 = \mathbb{E}_{\mathbf{r}} \left[\sum_{i=1}^{N} (y_i - \mathbf{w}^\top (\mathbf{r} \odot \mathbf{x}_i))^2 \right] = \sum_{i=1}^{N} \mathbb{E}_{\mathbf{r}} \left[(y_i - \mathbf{w}^\top (\mathbf{r} \odot \mathbf{x}_i))^2 \right]. \tag{5}$$

Apply the bias-variance decomposition $\mathbb{E}[Z^2] = \mathbb{V}[Z] + \mathbb{E}[Z]^2$, holding for any scalar random variable Z.

$$\mathbb{E}_{\mathbf{r}} \|\mathbf{y} - \mathbf{X}\mathrm{diag}(\mathbf{r})\mathbf{w}\|_2^2 = \sum_{i=1}^{N} \left[\mathbb{V}_{\mathbf{r}} \left[y_i - \mathbf{w}^\top (\mathbf{r} \odot \mathbf{x}_i) \right] + \left(\mathbb{E}_{\mathbf{r}} \left[y_i - \mathbf{w}^\top (\mathbf{r} \odot \mathbf{x}_i) \right] \right)^2 \right]. \tag{6}$$

The operator $\mathbb{V}_{\mathbf{r}}$ is invariant to deterministic translations. Therefore,

$$\mathbb{E}_{\mathbf{r}} \|\mathbf{y} - \mathbf{X}\mathrm{diag}(\mathbf{r})\mathbf{w}\|_2^2 = \sum_{i=1}^{N} \left[\mathbb{V}_{\mathbf{r}} \left[-\mathbf{w}^\top (\mathbf{r} \odot \mathbf{x}_i) \right] + \left(\mathbb{E}_{\mathbf{r}} \left[y_i - \mathbf{w}^\top (\mathbf{r} \odot \mathbf{x}_i) \right] \right)^2 \right]. \tag{7}$$

Once expanded the product $\mathbf{w}^\top (\mathbf{r} \odot \mathbf{x}_i)$ in components, we get

$$\mathbb{E}_{\mathbf{r}} \|\mathbf{y} - \mathbf{X}\mathrm{diag}(\mathbf{r})\mathbf{w}\|_2^2 = \sum_{i=1}^{N} \left[\mathbb{V}_{\mathbf{r}} \left[-\sum_{j=1}^{d} w_j r_j X_{ij} \right] + \left(\mathbb{E}_{\mathbf{r}} \left[y_i - \sum_{j=1}^{d} w_j r_j X_{ij} \right] \right)^2 \right] \tag{8}$$

For each i-th term of the summation, use the properties of variance and expected values with respect to linear combinations of independent random variables. This yields

$$\mathbb{E}_{\mathbf{r}}\|\mathbf{y}-\mathbf{X}\text{diag}(\mathbf{r})\mathbf{w}\|_2^2 = \sum_{i=1}^{N}\left[\sum_{j=1}^{d} w_j^2 X_{ij}^2 \mathbb{V}_{\mathbf{r}}\left[r_j\right] + \left(y_i - \sum_{j=1}^{d} w_j X_{ij}\mathbb{E}_{\mathbf{r}}\left[r_j\right]\right)^2\right] \tag{9}$$

$$= \sum_{i=1}^{N}\left[\sum_{j=1}^{d} w_j^2 X_{ij}^2 \cdot \theta(1-\theta) + \left(y_i - \sum_{j=1}^{d} w_j X_{ij}\cdot\theta\right)^2\right], \tag{10}$$

being the latter equality a direct consequence of the formulæ for the variance and the expected value for a $\text{Bernoulli}(\theta)$ distribution. Therefore, by highlighting the terms $\theta(1-\theta)$ and θ in front of the relative summations, we get

$$\mathbb{E}_{\mathbf{r}}\|\mathbf{y}-\mathbf{X}\text{diag}(\mathbf{r})\mathbf{w}\|_2^2 = \theta(1-\theta)\sum_{i=1}^{N}\sum_{j=1}^{d} w_j^2 X_{ij}^2 + \sum_{i=1}^{N}\left(y_i - \theta\sum_{j=1}^{d} X_{ij} w_j\right)^2. \tag{11}$$

By using the definition of Euclidean norm,

$$\mathbb{E}_{\mathbf{r}}\|\mathbf{y}-\mathbf{X}\text{diag}(\mathbf{r})\mathbf{w}\|_2^2 = \theta(1-\theta)\sum_{i=1}^{N}\sum_{j=1}^{d} w_j^2 X_{ij}^2 + \|\mathbf{y}-\theta\mathbf{X}\mathbf{w}\|_2^2 \tag{12}$$

Let us consider the first addend of (12) separately. By rearranging the summing ordering and replacing X_{ij}^2 with two identical copies of X_{ij}, we get

$$\sum_{i=1}^{N}\sum_{j=1}^{d} w_j^2 X_{ij}^2 = \sum_{j=1}^{d} w_j^2\left(\sum_{i=1}^{N} X_{ij}^2\right) = \sum_{j=1}^{d} w_j^2\left(\sum_{i=1}^{N} X_{ij}X_{ij}\right) \tag{13}$$

$$= \sum_{j=1}^{d} w_j^2\left(\sum_{i=1}^{N}(\mathbf{X})_{ji}^{\top}X_{ij}\right) = \sum_{j=1}^{d} w_j^2[\text{diag}(\mathbf{X}^{\top}\mathbf{X})]_{jj} \tag{14}$$

where we have exploited the transposition and the row-by-column product definitions. By squaring and square-rooting the second factor in the summation we obtain

$$\sum_{i=1}^{N}\sum_{j=1}^{d} w_j^2 X_{ij}^2 = \sum_{j=1}^{d} w_j^2([\text{diag}(\mathbf{X}^{\top}\mathbf{X})]_{jj}^{1/2})^2. \tag{15}$$

By noticing that the square-root of a diagonal matrix is a diagonal matrix whose entries are the square roots of the original entries, we obtain

$$\sum_{i=1}^{N}\sum_{j=1}^{d} w_j^2 X_{ij}^2 = \sum_{j=1}^{d} w_j^2([\text{diag}(\mathbf{X}^{\top}\mathbf{X})^{1/2}]_{jj})^2 = \sum_{j=1}^{d}(w_j[\text{diag}(\mathbf{X}^{\top}\mathbf{X})^{1/2}]_{jj})^2. \tag{16}$$

Apply the definition of row-by-column matrix product between a diagonal matrix and a vector.

$$\sum_{i=1}^{N}\sum_{j=1}^{d} w_j^2 X_{ij}^2 = \sum_{j=1}^{d}([\text{diag}(\mathbf{X}^{\top}\mathbf{X})^{1/2}\mathbf{w}]_j)^2 \tag{17}$$

$$= \|\text{diag}(\mathbf{X}^{\top}\mathbf{X})^{1/2}\mathbf{w}\|_F^2, \tag{18}$$

where, in (18), we used the definition of Frobenius norm. Replacing (18) in (12), leads to to prove (3).

In order to elicit (4), it is enough to notice that (15) can be rewritten as

$$\sum_{j=1}^{d} w_j^2 [\text{diag}(\mathbf{X}^\top \mathbf{X})]_{jj} = \sum_{j=1}^{d} w_j [\text{diag}(\mathbf{X}^\top \mathbf{X})]_{jj} w_j = \mathbf{w}^\top \text{diag}(\mathbf{X}^\top \mathbf{X}) \mathbf{w}, \tag{19}$$

being the last term equivalent to $\|\mathbf{w}\|_{\text{diag}(\mathbf{X}^\top \mathbf{X})}$, by exploiting the definition of norm induced by a symmetric and positive definite matrix [**?**]. Therefore, (19), once plugged into (15), leads to prove (4). This completes the proof. □

We can apply the identical analysis to the case of a deep neural network. In such a case, the input data matrix $\mathbf{X}$ is processed across subsequences of ℓ linear layers (represented by weights $\mathbf{W}^{(\ell)}$), with intermediate gating functions, pooling and feature normalization steps. Despite the latter non-linearities, since the values sampled from a Bernoulli are always either 0 or 1, it is enough to enumerate all the possible binary combinations of activations/inhibitions, accounting for the probability of their occurrence. This allows to retrieve (a different regularization term but) the same weighting factor $\theta(1-\theta)$.

Finally, let us note that, despite in the previous analysis the parameter θ was considered to be fixed, we can easily generalize it for $\theta = \theta(t)$. Precisely, it is enough to fix t arbitrary, apply the previous proofs by replacing θ with $\theta(t)$ and finally exploit the generality of t.

This concludes the theoretical analysis reported in the paper.

1.2. Curriculum Dropout, Curriculum Learning

In this Section, we justify the name Curriculum Dropoutby proving its equivalence to Curriculum Learning [1]. Precisely, with respect to the definition of a curriculum distribution provided in [1], we show that the latter is naturally induced by the proposed Curriculum Dropout. After recapping the definition of Curriculum Learning, we will detail how our approach naturally induces a curriculum distribution.

Definition of curriculum distribution. Within a classical machine learning algorithm, all training examples are presented to the model in an unordered manner, frequently applying a random shuffling. Actually, this is very different from what happens for the human training process, that is education. Indeed, the latter is highly structured so that the level of difficulty of the concepts to learn is proportional to the *age* of the people, managing easier knowledge when babies and harder when adults. This "start small" paradigm will likely guide the learning process [1].

Following the same intuition, [1] proposes to subdivide the training examples based on their difficulty. Then, the learning is configured so that easier examples come first, eventually complicating them and processing the hardest ones at the end of the training. This concept is formalized by introducing a learning time $\lambda \in [0, 1]$, so that training begins at $\lambda = 0$ and ends at $\lambda = 1$. At time λ, $Q_\lambda(z)$ denotes the distribution which a generic training example z is drawn from. The notion of curriculum learning is formalized requiring that Q_λ ensures a sampling of examples z which are easier than the ones sampled from $Q_{\lambda+\varepsilon}$, $\varepsilon > 0$. Mathematically, this is formalized by assuming

$$Q_\lambda(z) \propto W_\lambda(z) P(z). \tag{20}$$

In (20), $P(z)$ is the target training distribution, accounting for all examples, both easy and hard ones. The sampling from P is corrected by the factor $0 \leq W_\lambda(z) \leq 1$ for any λ and z. The interpretation for $W_\lambda(z)$ is the measure of the difficulty of the training example z. The maximal complexity for a training example is fixed to 1 and reached at the end of the training, *i.e.* $W_1(z) = 1$, *i.e.* $Q_1(z) = P(z)$. The weights $W_\lambda(z)$ must be chosen in such a way that

$$H(Q_\lambda) < H(Q_{\lambda+\varepsilon}), \tag{21}$$

where Shannon's entropy $H(Q_\lambda)$ models the fact that the quantity of information exploited by the model during training increases with respect to λ.

Curriculum dropout naturally induces a curriculum distribution. As clarified in the paper, let us consider dropout applied on the input layer only. In addition to make our analysis more understandable (see Fig. 1), this is not restrictive since we can apply the same arguments to any of the intermediate layer.

Let us denote $\mathcal{Z}_0$ the original dataset and assume to sample from it a d-dimensional image z_0 according to a distribution π. Clearly, the natural choice for π will be a uniform distribution. Moreover, here, we measure the dimensionality d of image by means of the total number of pixels.

Figure 1. From left to right, during training (red arrows), our curriculum dropout gradually increases the amount of Bernoulli multiplicative noise, generating multiple partitions (orange boxes) within the ***dataset*** (yellow frame) and the ***feature representation*** layers (not shown here). Differently, the original dropout [9, 15] (blue arrow) mainly focuses on the hardest partition only, complicating the learning from the beginning and potentially damaging the network classification performance.

While dropping out units in the input layer (*i.e.* pixels in z_0), we augment $\mathcal{Z}_0$ by adding all images in $\mathcal{Z}_0$ with *one* pixel se to zero (colored in black) and also all images in $\mathcal{Z}_0$ with *two* pixel se to zero and so on. This creates the dataset $\mathcal{Z}$, effectively used for dropout training, where any image $z \in \mathcal{Z}$ is obtained from an image $z_0 \in \mathcal{Z}_0$ by corrupting it through multiplicative Bernoulli noise. Equivalently, we can think about entrywise multiplying z_0 with a binary mask b. Therefore, we get

$$\mathbb{P}[\text{sampling } z] = \mathbb{P}[\text{sampling } z_0] \cdot \mathbb{P}[\text{sampling } b] = \pi(z_0) \cdot \mathbb{P}[\text{sampling } b]$$

In other words, any dropped out image z is uniquely determined by the original image z_0 and the binary mask b. One way to characterize that masks is by counting i, that is the number of zero entries of b. That leads to

$$\mathbb{P}[\text{sampling } z] = \pi(z_0) \cdot \binom{d}{i} (1-\theta)^i \theta^{d-i} \tag{22}$$

since b has entries set to zero (each realized with probability $1-\theta$) and the remaining set to one. The latter, being $d-i$ in total, are realized in correspondence of a success for the Bernoulli(θ) variable: therefore we obtain the term θ^{d-i}.

Let us introduce our curriculum function $\theta(t) = (1-\overline{\theta})\exp(-\gamma t) + \overline{\theta}$ (we will omit the pedix "$_{\text{curriculum}}$" for notational simplicity). Let us re-parametrize $t = \lambda T$ such that the training time (measured from 0 to the total number T of gradients updates) spans the range $[0, 1]$, starting at time $\lambda = 0$ and ending at time $\lambda = 1$. Therefore, by modifying (22), we introduce the following curriculum learning distribution

$$Q_\lambda(z) = \pi(z_0) \cdot \binom{d}{i} (1-\theta(\lambda T))^i \theta(\lambda T)^{d-i}. \tag{23}$$

Let us define

$$P(z) = Q_1(z). \tag{24}$$

When re-parametrizing $Q_\lambda(z) = Q_\lambda(z_0, i)$, we get a mixed distribution (discrete with respect to i and continous with respect to z_0). Hence,

$$\int Q_\lambda(z)dz = \int_{\mathcal{Z}_0} \pi(z_0)dz_0 \cdot \sum_{i=0}^{d} \binom{d}{i} (1-\theta(\lambda T))^i \theta(\lambda T)^{d-i} = 1 \tag{25}$$

because π is a normalized over its support $\mathcal{Z}_0$ and because the second factor equals one thanks to the Binomial Theorem.

If we compute the entropy of Q_λ, we obtain

$$H(Q_\lambda) = H(\text{Binomial}(d,\theta(\lambda T))) \cdot H(\pi), \tag{26}$$

being

$$H(\text{Binomial}(d,\theta(\lambda T))) = \frac{1}{2}\log[2\pi e d \cdot \theta(\lambda T)(1-\theta(\lambda T)] + O\left(\frac{1}{d}\right) \tag{27}$$

a strictly increasing function of λ. To see that, notice that it is enough to prove that $\theta(\lambda T)(1-\theta(\lambda T))$ is increasing as a function of λ. But, this is true since composition of the composition of strictly decreasing functions is strictly increasing. Precisely, the two functions to be composed are $\theta(\lambda T)$ and $f(x) = x(1-x)$, both of them strictly decreasing. Indeed,

$$\theta'(\lambda T) = -\gamma T(1-\overline{\theta})\exp(-\gamma \lambda T) < 0$$

for any λ and

$$f'(x) = 1 - 2x < 0$$

since we evaluate $f(\theta(\lambda T))$ and $\theta(\lambda T) > \overline{\theta} \geq 1/2$ for any λ. Therefore, for any $\varepsilon > 0$,

$$H(Q_\lambda) < H(Q_{\lambda+\varepsilon}).$$

This completes the proof.

2. Experimental setup

In this section, we detail the network architectures and hyperparameters used for the experiments, and provide some more extensive results.

2.1. Gamma

As claimed in footnote 2, we here show that Curriculum Dropout with exponential scheduling, is very robust against the choice of the decaying factor γ. Namely, *any curriculum always leads to better generalization than the no-curriculum strategy* (*i.e.* the standard dropout scheme). Moreover, the heuristic $\gamma = 10/T$ defined in the paper is proved to be an effective rule. Results are reported in Table 1 for the Cifar-10 dataset. The architecture and experimental setup are as in section 2.2.

Dataset	Dropout	$\gamma = 10^{-3}$	$\gamma = 7 \times 10^{-4}$	$\gamma = 3 \times 10^{-4}$	$\gamma = 10^{-4}$
CIFAR-10	73.29	73.52	73.87	**73.99**	73.69

Table 1. Accuracies on CIFAR-10 with regular Dropout and Curriculum Dropout with different values of the decaying factor γ. Best result (bold), corresponds to the heuristic proposed in the paper, *i.e.* $\gamma = 10/T$. In fact, we train the network on 50000 samples for 80 epochs, with batch-size of 128. This corresponds to $T \approx 3.1 \times 10^4$ iterations and yields $\gamma T \approx 10$.

2.2. Network Architectures

Layer Type	Layer Size	Filter Size	Padding/Stride
conv	64 filters	3x3	1/1
max pool		2x2	0/2
conv	128 filters	3x3	1/1
max pool		2x2	0/2
fc	1024 units		
fc	1024 units		
softmax	# of classes		

Table 2. Network architecture for CIFAR-10 and CIFAR-100. The only difference is the size of the softmax layer.

Layer Type	Layer Size	Filter Size	Padding/Stride
conv	96 filters	5x5	2/1
max pool		3x3	0/2
conv	128 filters	5x5	2/1
max pool		3x3	0/2
conv	256 filters	5x5	2/1
max pool		3x3	0/2
fc	2048 units		
fc	2048 units		
softmax	# of classes		

Table 3. Network architecture for SVHN, Cifar, Caltech-101 and Caltech-256. The only difference is the size of the softmax layer.

Layer Type	Layer Size	Filter Size	Padding/Stride
conv	32 filters	5x5	1/1
max pool		2x2	0/2
conv	48 filters	5x5	1/1
max pool		2x2	0/2
fc	2048 units		
fc	1024 units		
softmax	10 units		

Table 4. Network architecture for MNIST.

Tables 2, 3, 4 show the network architectures used for different experiments. Relu Learning rate and momentum were set to 0.0001 (0.001 for experiments in CIFAR-10 and CIFAR-100) and 0.95, respectively, for each run. Gamma was set following the heuristics reported in the manuscript. We initialized weights with normals with std-dev 0.01. Since all architectures rely on ReLU activations, units are also initialized with a small positive bias $b = 0.01$ in order to avoid dead neurons.

2.3. Full Results

Table 5 is an extended version of the one in the manuscript, in that it shows mean accuracies together with standard deviations. To calculate the mean accuracies, we selected the 10 highest accuracy values for each of the 10 runs, calculated the average of these values and then calculated the average (and the standard deviation) over the 10 mean values.

Dataset	Architecture	Configuration (n or $n\bar{\theta}$ fixed)	Classes	Unregularized network	Dropout [9, 15]	Anti-Curriculum	*Curriculum Dropout*
MNIST [11]	MLP	n	10	98.67 ± 0.06	99.05 ± 0.03	98.76 ± 0.04	99.02 ± 0.00
	CNN-1	n		99.25 ± 0.03	99.39 ± 0.01	99.20 ± 0.05	99.43 ± 0.01
Double MNIST	CNN-2	n	55	92.48 ± 0.82	93.91 ± 0.68	93.21 ± 0.80	94.83 ± 0.50
	CNN-2	$n\bar{\theta}$			93.36 ± 0.73	93.01 ± 0.56	93.60 ± 0.82
SVHN [12]	CNN-2	n	10	84.63 ± 0.40	86.98 ± 0.16	85.80 ± 0.16	87.28 ± 0.20
	CNN-2	$n\bar{\theta}$			86.22 ± 0.22	86.14 ± 0.15	86.69 ± 0.19
CIFAR-10 [10]	CNN-1	n	10	73.06 ± 0.15	73.29 ± 0.33	72.38 ± 0.24	73.69 ± 0.28
CIFAR-100 [10]	CNN-1	n	100	39.70 ± 0.21	40.71 ± 0.05	39.70 ± 0.24	41.36 ± 0.32
Caltech-101 [6]	CNN-2	n	101	28.56 ± 0.68	32.78 ± 0.87	30.13 ± 1.31	33.28 ± 0.77
Caltech-256 [7]	CNN-2	n	256	14.39 ± 0.64	16.75 ± 0.42	14.18 ± 0.24	17.62 ± 0.98

Table 5. Comparison of the proposed scheduling versus [9, 15].

References

[1] Y. Bengio, J. Louradour, R. Collobert, and J. Weston. Curriculum learning. In *ICML*, 2009. 2, 4

[2] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein. Distributed optimization and statistical learning via the alternating direction method of multipliers. *Found. Trends Mach. Learn.*, 3(1):1–122, 2011. 2

[3] J. Cavazza and V. Murino. Active Regression with Adaptive Huber loss. In *CoRR:1606.01568*, 2016. 2

[4] K. Crammer, A. Kulesza, and M. Dredze. Adaptive regularization of weight vectors. In *NIPS*. 2009. 2

[5] T. Evgeniou, M. Pontil, and T. Poggio. Regularization networks and support vector machines. *Advances in Computational Mathematics*, 13(1), 2000. 2

[6] L. Fei-Fei, R. Fergus, and P. Perona. Learning generative visual models from few training examples: An incremental bayesian approach tested on 101 object categories. In *CVPR workshop*, 2004. 7

[7] G. Griffin, A. Holub, and P. Perona. Caltech-256 object category dataset. In *Technical Report 7694, California Institute of Technology*, 2007. 7

[8] L. Hansen and C. Rasmussen. Pruning from adaptive regularization. *Neural Computation*, 6(6):1222–1231, 1994. 2

[9] G. E. Hinton, N. Srivastava, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov. Improving neural networks by preventing co-adaptation of feature detectors. In *CoRR:1207.0580*, 2012. 2, 5, 7

[10] A. Krizhevsky and G. Hinton. Learning multiple layers of features from tiny images. *Master's thesis, Department of Computer Science, University of Toronto*, 2009. 7

[11] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. In *Proceedings of the IEEE, 86(11):22782324*, 2009. 7

[12] Y. Netzer, T. Wang, A. Coates, A. Bissacco, B. Wu, and A. Y. Ng. Reading digits in natural images with unsupervised feature learning. In *NIPS workshop*, 2011. 7

[13] W. Rudin. *Real and Complex Analysis, 3rd Ed.* McGraw-Hill, Inc., 1987.

[14] M. Soltanolkotabi, E. Elhamifar, and E. J. Candès. Robust subspace clustering. In *CoRR:1301.2603*, 2013. 2

[15] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov. Dropout: A simple way to prevent neural networks from overfitting. *J. Mach. Learn. Res.*, 15(1):1929–1958, 2014. 2, 5, 7