Dynamic Loss

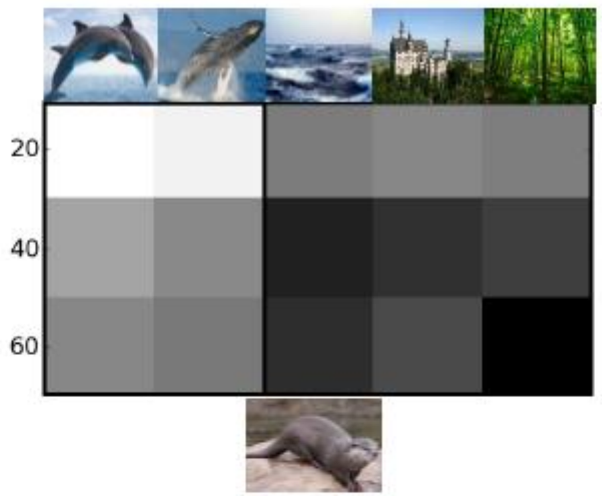
赵学亮

Outline

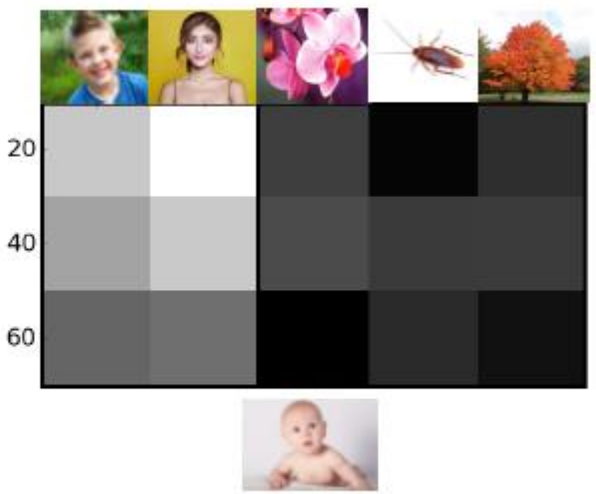
- Learning to Teach with Dynamic Loss Functions
  - Dynamic loss
  - Method
  - Experiment

- Addressing the Loss-Metric Mismatch with Adaptive Loss Alignment
  - Adaptive loss alignment(ALA)
  - Method
  - Experiment

Dynamic Loss（NIPS, 2018）



(a) Class *Otter*



(b) Class *Baby*

$$l_{ce}(f_w(x), y) = -y^{\mathrm{T}} \log f_w(y|x)$$

$$\Downarrow$$

$$l_{\Phi_t}(f_w(x), y) = -\sigma(y^{\mathrm{T}} \Phi_t \log f_w(y|x)).$$

Chanllenge

- The task-specific objective is usually non-smooth w.r.t. student model
- The final evaluation of the student model is incurred on the dev set, disjoint with the training dataset where the teaching process actually happens

Student Model

$$\min_{\omega \in \Omega} \sum_{(x,y) \in D_{train}} l(f_\omega(x), y).$$

$$L(f_\omega, D) = \sum_{(x,y) \in D} l(f_\omega(x), y)$$

$$\omega_{t+1} = \omega_t - \eta_t \frac{\partial L_\Phi(f_{\omega_t}, D_{train}^t)}{\partial \omega_t}$$

Teacher Model

$$\omega_{t+1} = \omega_t - \eta_t \frac{\partial L_{\Phi_t}(f_{\omega_t}, D^t_{train})}{\partial \omega_t} = \omega_t - \eta_t \frac{\partial L_{\mu_\theta(s_t)}(f_{\omega_t}, D^t_{train})}{\partial \omega_t}$$

$$f_{\omega^*} = \mathcal{F}(D_{train}, \mu_\theta).$$

$$\max_\theta \mathcal{M}(f_{\omega^*}, D_{dev}) = \max_\theta \mathcal{M}(\mathcal{F}(D_{train}, \mu_\theta), D_{dev}).$$

Challenge1

- Smooth the task-specific measure to its expected version where the expectation is taken on the direct output of student model.

$$\tilde{m}(f_\omega(x), y) = \sum_{y^* \in \mathcal{Y}} m(y^*, y) p_\omega(y^* | x),$$

$$\frac{\partial \tilde{m}(f_\omega(x), y)}{\partial \omega} = \sum_{y^* \in \mathcal{Y}} m(y^*, y) \frac{\partial p_\omega(y^* | x)}{\partial \omega}.$$

Challenge2

- View the sequential process of student model optimization as a special feed-forward process of a deep neural network where each t corresponds to one layer
- RMD corresponds to the backpropagation process looping the SGD process backwards from T

Challenge2

$$d\omega_T = \frac{\partial \tilde{\mathcal{M}}(f_{\omega_T}, D_{dev})}{\partial \omega_T} = \sum_{(x,y)\in D_{dev}} \frac{\partial \tilde{m}(f_{\omega_T}(x), y)}{\partial \omega_T}.$$

$$d\omega_t = \frac{\partial \tilde{\mathcal{M}}(f_{\omega_t}, D_{dev})}{\partial \omega_t} = d\omega_{t+1} - \eta_t \frac{\partial^2 L_{\mu_\theta(s_t)}(f_{\omega_t}, D^t_{train})}{\partial \omega_t^2} d\omega_{t+1}.$$

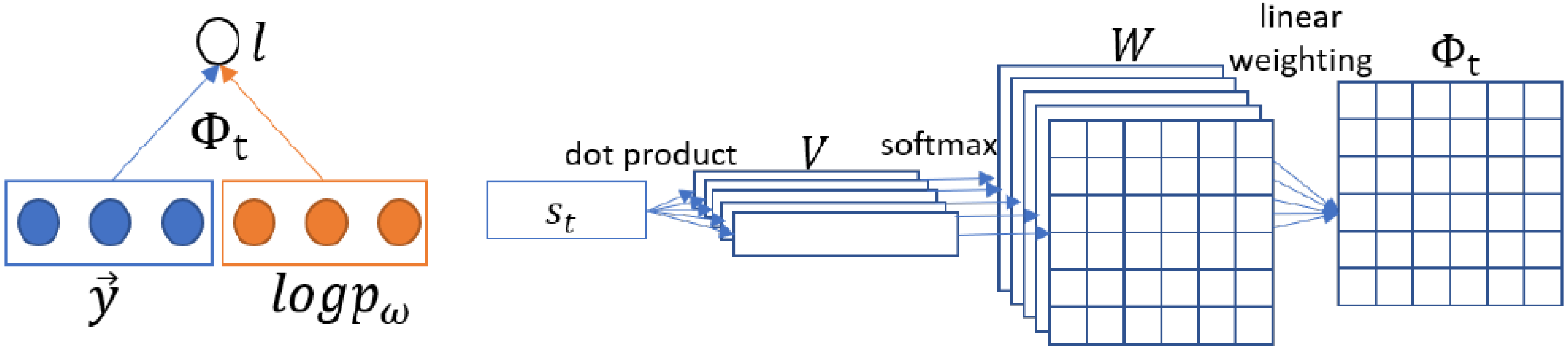
$$d\theta = d\theta - \eta_t \frac{\partial^2 L_{\mu_\theta(s_t)}(f_{\omega_t}, D^t_{train})}{\partial \theta \partial \omega_t} d\omega_{t+1}.$$

Algorithm

Algorithm 1 Training Teacher Model μ_θ

Input: Continuous relaxation $\tilde{m}$. Initial value of θ.
while Teacher model parameter θ not converged **do** ▷ One *teacher optimization step*
  Randomly initialize student model parameter ω_0.
  for each time step $t = 0, \cdots, T-1$ **do** ▷ Teach student model
    Conduct student model training step via Eqn. (6).
  end for
  $d\theta = 0$. Compute $d\omega_T$ via Eqn. (3).
  for each time step $t = T-1, \cdots, 0$ **do** ▷ Reversely calculating the gradient $d\theta$
    Update $d\theta$ as Eqn. (5).
    Compute $d\omega_t$ as Eqn. (7).
  end for
  Update θ using $d\theta$ via gradient based optimization algorithm.
end while
Output: the final teacher model μ_θ.

Image Classification



(a) loss function (b) teacher model

$$l_{ce}(f_w(x), y) = -\bar{y}^{\mathrm{T}} \log f_w(y|x)$$

$$l_{\Phi_t}(f_w(x), y) = -\sigma(y^{\mathrm{T}} \Phi_t \log f_w(y|x))$$

Image Classification

Table 1: The recognition results (error rate %) on MNIST dataset.

Student Model/ Loss	Cross Entropy [11]	Smooth [41]	Large-Margin Softmax [37]	L2T-DLF
MLP	1.94	1.89	1.83	**1.69**
LeNet	0.98	0.94	0.88	**0.77**

Table 2: The recognition results (error rate %) on CIFAR-10 (C10) and CIFAR-100 (C100) dataset

Student Model/ Loss	Cross Entropy [11]	Smooth [41]	Large-Margin Softmax [37]	L2T-DLF
	C10/C100	C10/C100	C10/C100	C10/C100
ResNet-8	12.45/39.79	12.08/39.52	11.34/38.93	**10.82/38.27**
ResNet-20	8.75/32.33	8.53/32.01	8.02/31.65	**7.63/30.97**
ResNet-32	7.51/30.38	7.42/30.12	7.01/29.56	**6.95/29.25**
WRN	3.80/-	3.81/-	3.69/-	**3.42/-**
DenseNet-BC	3.54/-	3.48/-	3.37/-	**3.08/-**

Image Classification

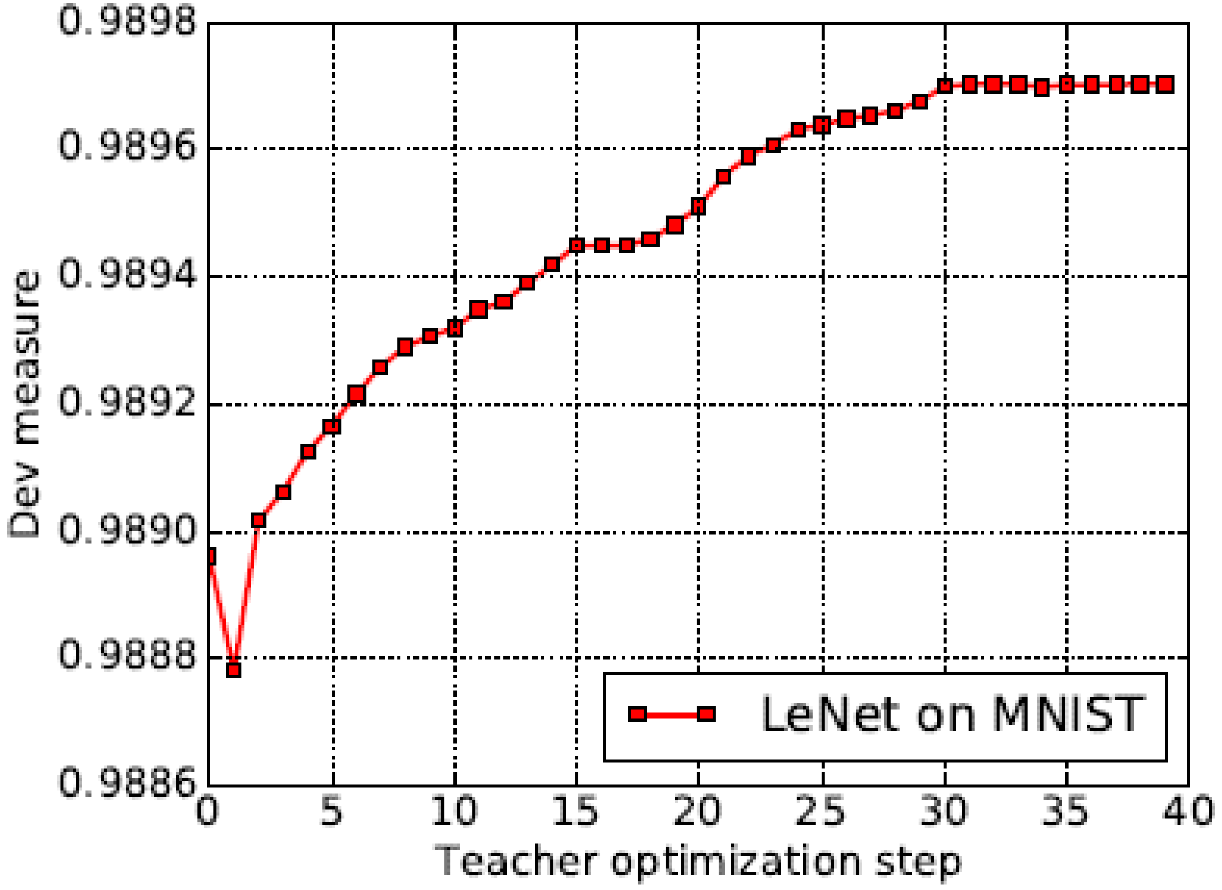
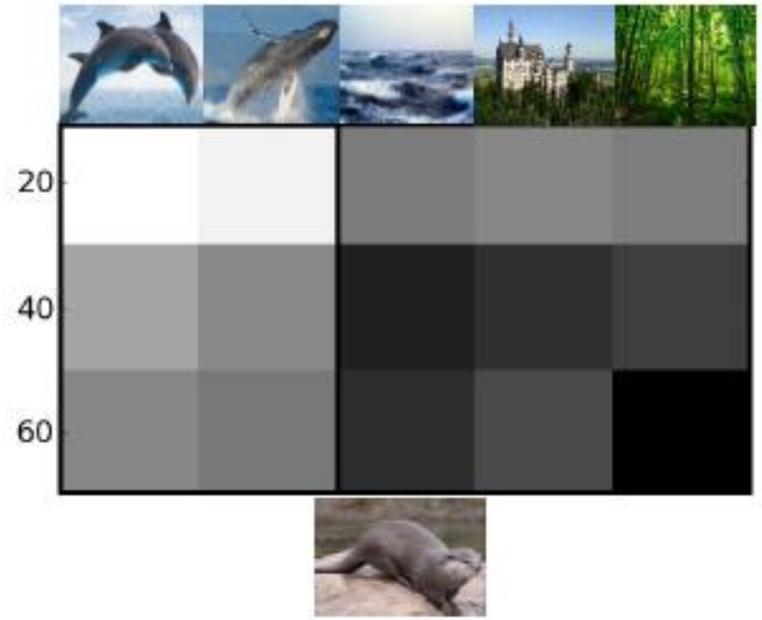


Image Classification



(a) Class *Otter*

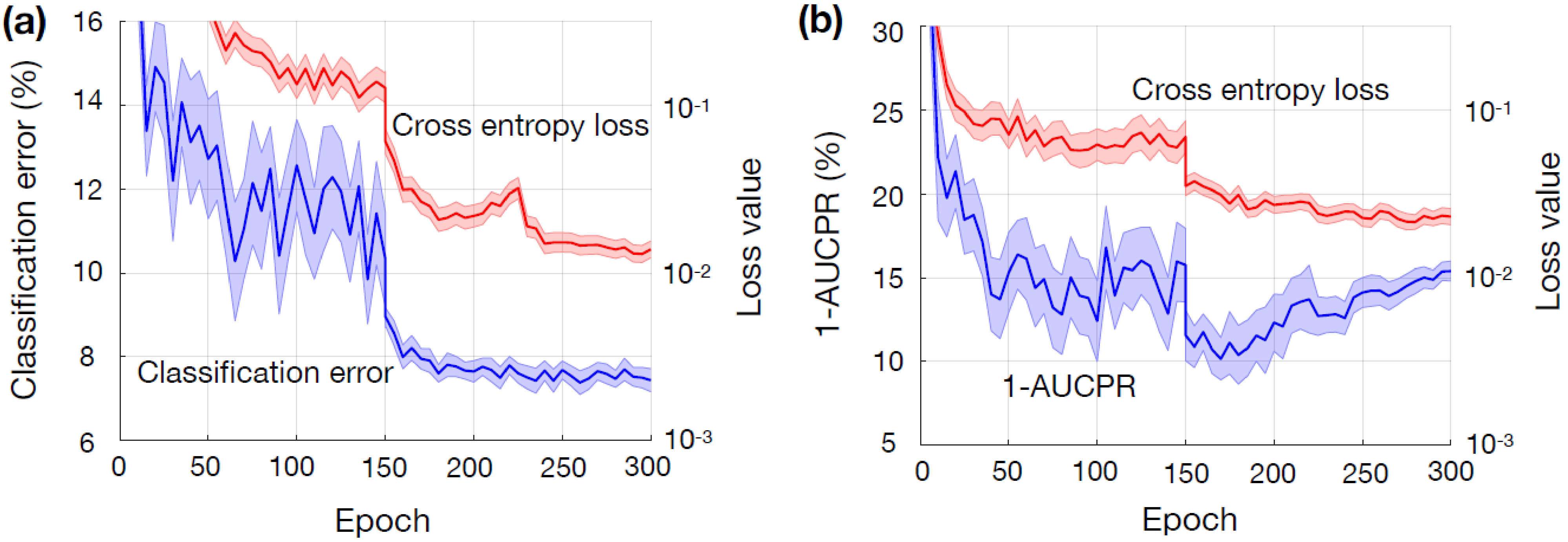
(b) Class *Baby*

Neural Machine Translation

Table 3: The translation results (BLEU score) on IWSLT-14 German-English task.

Student Model/ Loss	Cross Entropy [55]	RL [44]	AC [3]	Softmax-Margin [12]	L2T-DLF
LSTM-1	27.28	27.53	27.75	28.12	**29.52**
LSTM-2	30.86	31.03	31.21	31.22	**31.75**
Transformer	34.01	34.32	34.34	34.46	**34.80**

Adaptive Loss Alignment（ICML, 2019）



Loss-metric mismatch on CIFAR-10

Loss Functions with Parameters

$$l_{ce}(f_w(x), y) = -y^{\mathrm{T}} \log f_w(y|x)$$

$$l_{\Phi_t}(f_w(x), y) = -\sigma(y^{\mathrm{T}} \Phi_t \log f_w(y|x))$$

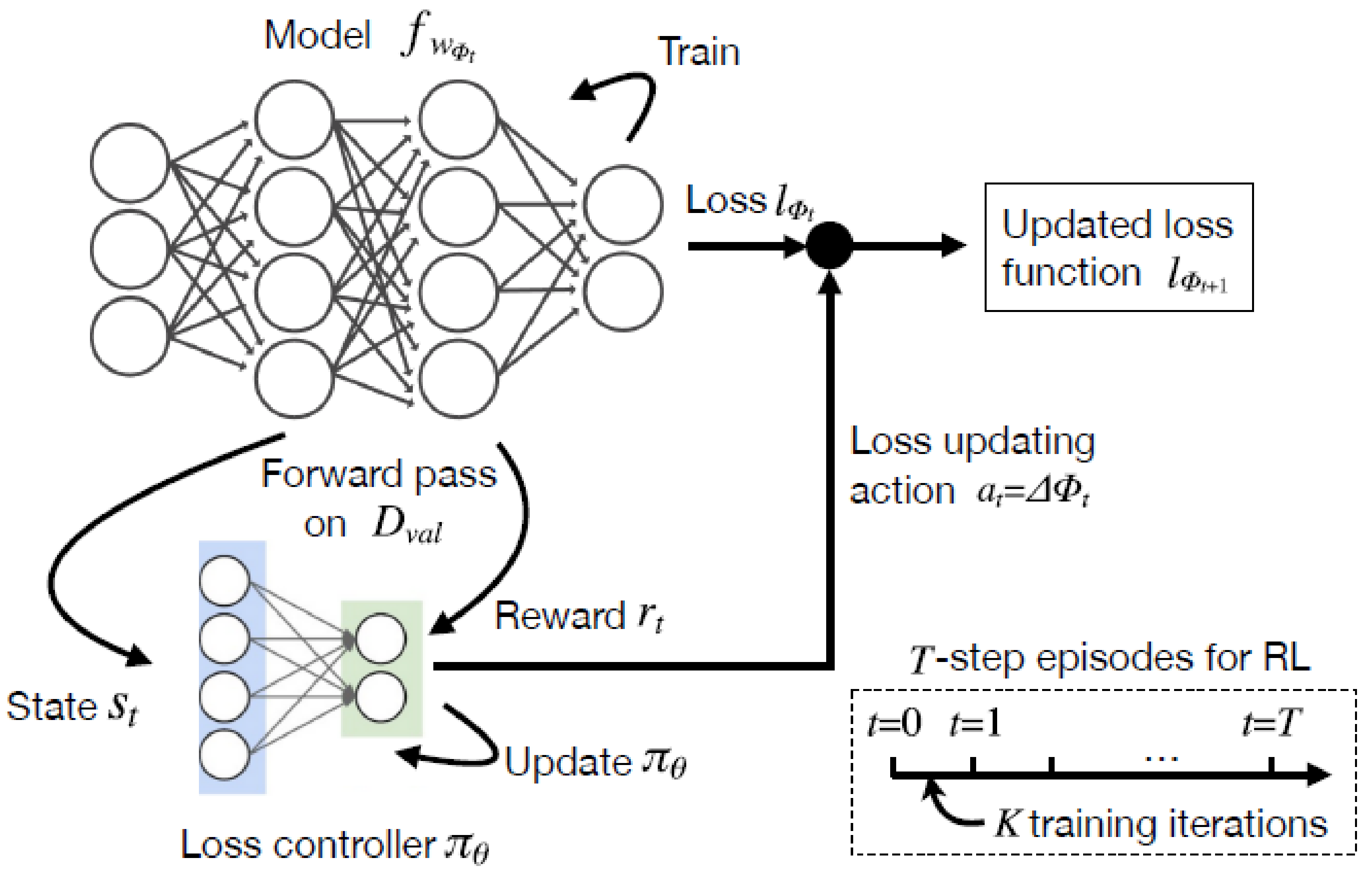
Problem Formalization

- Improve evaluation metric $M(f_w, D_{val})$ on validation set.
- By solving $min_w \sum_{(x,y) \in D_{train}} l(f_w(x), y)$
- Remedy: alternate direction optimization problem

$$\min_{\Phi} \quad \mathcal{M}(f_{w_\Phi}, D_{val}),$$

$$s.t.\ w_\Phi = \arg\min_{w} \sum_{(x,y) \in D_{train}} l_\Phi(f_w(x), y),$$

Reinforcement Learning



Reward:

$$\mathcal{M}_{t+1} = \sum_{j=1}^{K} \gamma^{K-j} \mathcal{M}(f_{wj}, D_{val}),$$

$$r_t = \mathrm{sign}(\mathcal{M}_t - \mathcal{M}_{t+1}),$$

Action: For every element Φ_t, sample action $a_t(i)$ from $\{-\beta, 0, \beta\}$

State: use the validation statistics to capture model training states

Learning Algorithm

Algorithm 1 Reinforcement Learning for ALA

Initialize each child model with random weights w
Initialize loss controller π_θ with random weights θ
Initialize loss parameters Φ_0 properly for a given task
Initialize replay memory $\mathcal{D}$
while not converged **do**
  for each state s_t **do**
    Sample action $a_t \sim \pi_\theta(a_t|s_t)$
    Take action a_t to update loss function to $l_{\Phi_{t+1}}$
    Update w by K SGD iterations with $l_{\Phi_{t+1}}$
    Collect reward r_t (Equation 5) and new state s_{t+1}
    Store $\langle s_t, a_t, r_t, s_{t+1} \rangle$ from all child models in $\mathcal{D}$
    Sample random experiences from $\mathcal{D}$
    Update θ to maximize reward via Equation 3
  end for
end while

$$\nabla_\theta J(\theta) = \mathbb{E}_\tau \left[\sum_{t=0}^{T} \nabla_\theta \log \pi_\theta(a_t|s_t) \sum_{k=t}^{T} (r_k - b_k) \right], \quad (3)$$

Instantiation on Classification

$$l_{\Phi_t}(f_w(x), y) = -\sigma(y^{\mathrm{T}} \Phi_t \log f_w(y|x)),$$

Confusion Matrix:

$$C_{i,j} = \frac{\sum_{d=1}^{|D_{val}|} -I(y_d, i) \log f^j_{w_{\Phi_t}}(x_d)}{\sum_{d=1}^{|D_{val}|} I(y_d, i)}$$

$$[C_{i,j}, C_{j,i}] \xrightarrow{\text{Policy Network}} \Phi_t(i,j) \text{ and } \Phi_t(j,i)$$

Different class pairs share the same controller !

Instantiation on Metric Learning

$$l_{tri}(f_w(x_{i,i^+,i^-})) = \max\left(0, F(d^+) - F(d^-) + \eta\right)$$

Distance mixture

Focal weighting

$$l_{\Phi_t} = \sum_{i=1}^{5} \Phi_t(i) F_i^+(d^+) + \sum_{i=1}^{5} \Phi_t(i+5) F_i^-(d^-),$$

$$l_{\Phi_t} = \frac{1}{\Phi_t(1)} \log\left[1 + \sum_{i+} \exp\left(\Phi_t(1) \cdot (d_{i+}^+ - \alpha)\right)\right] + \frac{1}{\Phi_t(2)} \log\left[1 + \sum_{i-} \exp\left(-\Phi_t(2) \cdot (d_{i-}^- - \alpha)\right)\right],$$

Results

Table 1. Classification error (%) on CIFAR-10 dataset. 10-run average and standard deviation are reported for ALA.

Method	ResNet-32	WRN	DenseNet
cross-entropy	7.51	3.80	3.54
Self-paced (Kumar et al., 2010)	7.47	3.84	3.50
L-Softmax (Liu et al., 2016)	7.01	3.69	3.37
L2T (Fan et al., 2018)	7.10	-	-
L2T-DLF (Wu et al., 2018)	6.95	3.42	3.08
ALA (random matrix Φ_t)	8.23±0.41	4.69±0.28	4.15±0.33
ALA (confusion matrix Φ_t)	7.42±0.04	3.74±0.02	3.55±0.02
ALA (single-network)	6.85±0.09	3.39±0.04	3.03±0.04
ALA (multi-network)	**6.79±0.07**	**3.34±0.04**	**3.01±0.02**

Table 2. AUCPR (%) on CIFAR-10 dataset. All methods use the same model architecture as adopted in (Eban et al., 2017). 10-run average and standard deviation are reported as ALA result.

Method	AUCPR
cross-entropy loss for optimizing accuracy	84.6
Pairwise AUCROC loss (Rakotomamonjy, 2004)	94.2
AUCPR loss (Eban et al., 2017)	94.2
ALA	**94.9±0.14**

Table 3. Recall(%)@k on Stanford Online Products dataset.

k	1	10	100	1000
Triplet (Schroff et al., 2015)	66.7	82.4	91.9	-
Margin (Wu et al., 2017)	72.7	86.2	93.8	98.0
BIER (Opitz et al., 2017)	72.7	86.5	94.0	98.0
HTL (Ge et al., 2018)	74.8	88.3	94.8	98.4
ABE-8 (Kim et al., 2018)	76.3	88.4	94.8	98.2
Triplet + ALA (Distance mixture)	75.7	89.4	95.3	98.6
Margin + ALA (Distance mixture)	**78.9**	**90.7**	**96.5**	**98.9**
Margin + ALA (Focal weighting)	77.9	90.1	95.8	98.7
Margin + ALA (FR policy transfer)	75.2	89.2	94.9	98.4

Table 4. Policy transfer from CIFAR-10 to ImageNet. Top-1 and Top-5 accuracy rates (%) are reported on ImageNet.

Method	Top-1	Top-5
RMSProp	73.5	91.5
PowerSign-cd (Bello et al., 2017)	73.9	91.9
RMSProp + ALA (CIFAR policy transfer)	74.3	92.1
RMSProp + ALA	**74.6**	**92.6**

Analysis

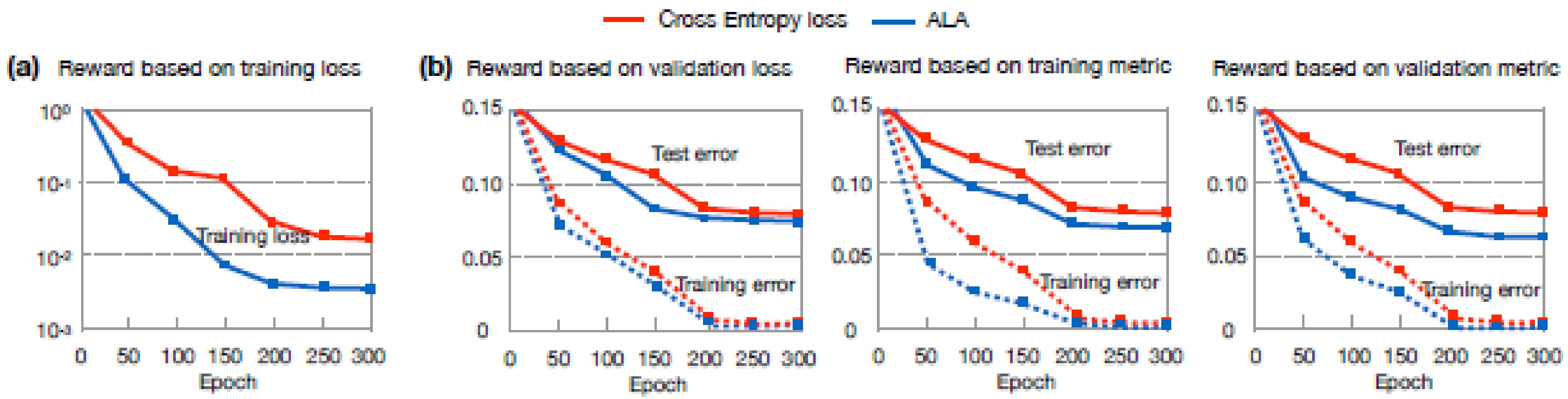


Figure 3. Analysis of optimization vs. generalization on CIFAR-10. (a) Comparing optimization performance in terms of the raw cross-entropy loss outputs on training data: Here ALA is rewarded by the training loss, and we observe that the measured training loss is consistently lower compared to the fixed cross-entropy loss, indicating improved optimization. (b) Comparing ALA policies trained with different rewards: the validation loss-based reward improves both optimization (training error) and generalization (test error), and the gains are larger when using the validation metric-based reward. In contrast, using the training metric as the reward yields smaller gains in test error, potentially due to the diminishing reward (error approaching zero) in training data.

Analysis

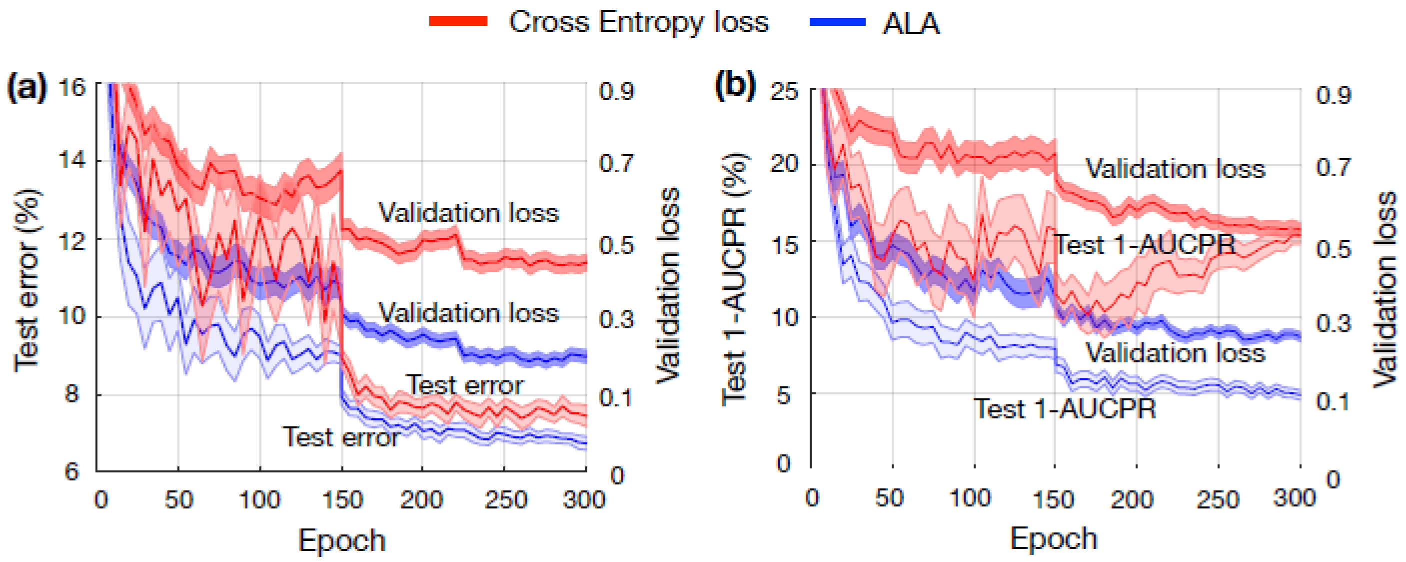


Figure 5. Validation loss vs. test metric of (a) classification error and (b) AUCPR on CIFAR-10. Curves are means over 10 runs initialized with different random seeds, and shaded areas show standard deviations. ALA uses the default reward based on the validation error metric, and improves both validation loss and test metric by addressing the loss-metric mismatch (both before and after the loss/metric drop due to learning rate change).

谢谢