Temperature in Cosine-based Softmax Loss

Supplementary Material

5. Softmax Representation

Softmax is derived from the optimization of

$$\min_{\boldsymbol{\alpha}} \frac{1}{\kappa}\sum_{c=1}^{C}\alpha_c\log\alpha_c - \sum_{c=1}^{C}\alpha_c z_c, \tag{16}$$

$$s.t. \sum_{c=1}^{C}\alpha_c = 1, \tag{17}$$

$$\alpha_c \geq 0\,\forall c\in\{1,\cdots,C\}, \tag{18}$$

as follows.

By introducing Lagrange multipliers λ and $\{\beta_c\}_{c=1}^C$ for the constraints (17, 18), the above optimization leads to

$$L = \frac{1}{\kappa}\sum_c\alpha_c\log\alpha_c - \sum_c\alpha_c z_c + \lambda(\sum_c\alpha_c - 1) - \sum_c\beta_c\alpha_c, \tag{19}$$

which produces the following derivatives and KKT conditions;

$$\frac{\partial L}{\partial\alpha_c} = \frac{1}{\kappa}(\log\alpha_c + 1) - z_c + \lambda - \beta_c = 0, \tag{20}$$

$$\frac{\partial L}{\partial\lambda} = \sum_c\alpha_c - 1 = 0, \tag{21}$$

$$\text{KKT: } \alpha_c \geq 0,\ \beta_c \geq 0,\ \beta_c\alpha_c = 0,\ \forall c. \tag{22}$$

From (20), we can derive

$$\alpha_c = \exp(\kappa(z_c - \lambda + \beta_c) - 1) > 0, \tag{23}$$

which is also accompanied by $\beta_c = 0$ in (22). On the other hand, λ can be determined so that (21) holds, finally resulting in the optimizer of softmax representation as

$$\alpha_c = \frac{\exp(\kappa z_c)}{\sum_k\exp(\kappa z_k)}. \tag{24}$$

5.1. Connection to Least-square approach

As shown in (8), the least-square representation is written in

$$\min_{\boldsymbol{\alpha}\in\Omega}\frac{1}{2}\boldsymbol{\alpha}^\top\tilde{\boldsymbol{W}}^\top\tilde{\boldsymbol{W}}\boldsymbol{\alpha} - \sum_c\alpha_c z_c. \tag{25}$$

In the case that the classifiers are less correlated, implying $\tilde{\boldsymbol{W}}^\top\tilde{\boldsymbol{W}} \approx \boldsymbol{I}$ (identity matrix) such as after sufficient training, (25) is further reduced to

$$\min_{\boldsymbol{\alpha}\in\Omega}\frac{1}{2}\sum_c\alpha_c^2 - \sum_c\alpha_c z_c. \tag{26}$$

Compared to (16), only difference is found in the first term which injects regularization about sparsity; that is, negative entropy and L_2-norm are introduced for (moderately) smoothing the coefficients $\boldsymbol{\alpha}$ in the minimization of (16) and (26), respectively. This analogy is a theoretical motivation to regard the softmax representation (9) as an (approximated) κ-parameterized optimizer for the least-square problem (25).

6. Analysis about κ

6.1. Characteristics

As the least-square formulation (8) is equivalent to one-class SVM [47], it produces *sparse* coefficients $\boldsymbol{\alpha}$ which contain a few numbers of non-zero elements corresponding to support vectors. It accordingly demands the softmax (9) to be also sparse as an approximation of $\boldsymbol{\alpha}$ (Section 5).

When the feature vector $\tilde{\boldsymbol{x}}$ is apart from the classifiers especially at an early stage of training, the logits are quite small; we can empirically observe that $\max_c|z_c| = \epsilon \ll 1$ for the immature features. For ease of discussion, suppose that we have m-prominent logits of $z_{c^*} = \epsilon$ and the other logits are zeros. As describe above, it is necessary to convert the less discriminative logits to m-sparse softmax of

$$p(\boldsymbol{z}) = \Big[\underbrace{\frac{1-\eta}{m},\cdots,\frac{1-\eta}{m}}_{m},\underbrace{\frac{\eta}{C-m},\cdots,\frac{\eta}{C-m}}_{C-m}\Big], \tag{27}$$

with small fraction $\eta \ll 1$. It is achieved by setting κ as

$$p(z_{c^*}) = \frac{1-\eta}{m} = \frac{\exp(\kappa\epsilon)}{m\exp(\kappa\epsilon) + (C-m)} \tag{28}$$

$$\Rightarrow \kappa = \frac{1}{\epsilon}[\log\frac{1-\eta}{\eta} + \log\frac{C-m}{m}] \gg 1. \tag{29}$$

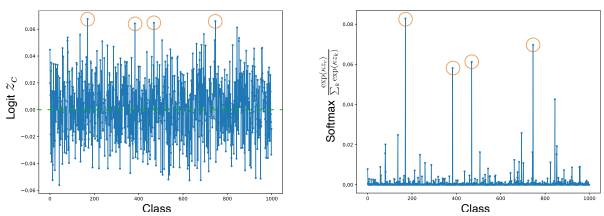
Thus, as shown in Figure 1, the LS-optimized κ^* is larger for immature features at early training epochs; Figure 5 demonstrates an empirical case for the feature at the 1st epoch which renders $\kappa^* = 102$ in our LS method. Then, as the training proceeds, the feature vector $\tilde{\boldsymbol{x}}$ approaches the classifier $\tilde{\boldsymbol{w}}_y$, producing discriminative logits with an enlarged ϵ to reduce κ in constructing sparse softmax.

6.2. Number of classes

Our method adaptively copes with various number of classes, C. Figure 6 summarizes the optimized κ^* over various C on diverse datasets. It is noteworthy that our method optimizes κ based on a feature representation and the number of classifiers (C), and generally speaking, the

Table 8. Training parameters.

	Table 2		Table 4b	Table 5	Table 6	Table 7
	Cifar-10/100	Food-101/ImageNet	Fine-tuning	Cosify	MS1M-RetinaFace	ImageNet-LT/iNat2018/Places-LT
Epochs	240	100	60	30	25	100 (1st) / 30 (2nd)
Learning rate	0.1	0.1	0.1($\boldsymbol{W}$), 0.001(Θ)	0.01	0.2	0.2
schedule	cosine	cosine	cosine	cosine	polynomial ($p=2$)	cosine
Weight decay	0.0005	0.0001	0.0001	0.0001	0.0005	0.0001
Batch size	128	256	128	256	512	256 (Random (1st) / Class-balanced (2nd) sampling)



(a) logits z_c (b) softmax $\frac{\exp(\kappa^* z_c)}{\sum_k \exp(\kappa^* z_k)}$

Figure 5. An example of an immature feature $\tilde{\boldsymbol{x}}$ by ResNet-50 at the 1st epoch on ImageNet training. The logits $z_c = \tilde{\boldsymbol{w}}_c^T \tilde{\boldsymbol{x}}$ are shown in (a) and are converted to softmax in (b) with the LS-optimized $\kappa^* = 102$. The points of higher logit scores, indicated by circles, win the sparse softmax weights via the larger κ^*.

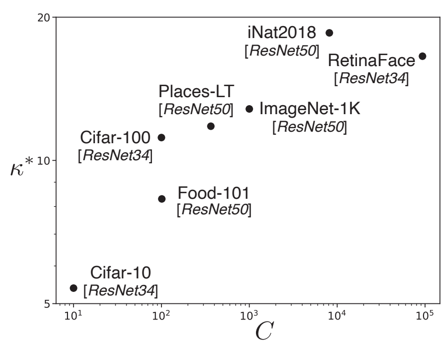


Figure 6. Relationship between the number of class (C) and the optimized κ^*.

larger number (C) of classifiers well describe an input feature in (**10**), enlarging κ^*.

6.3. Candidate values

In Section 2.3, we provide a candidate set over which the optimal κ is searched in a simple greedy manner. In this work, the set is simply composed of 20 values equally-log-spaced in $[e^{-2}, e^5]$ as

$$\kappa_j = \exp\left\{-2 + \frac{5-(-2)}{19} j\right\}, \; j \in \{0, \cdots, 19\}, \quad (30)$$

or practically written in PyTorch style by

$$\texttt{torch.exp(torch.linspace}(-2, 5, 20)). \quad (31)$$

7. Training procedure

Deep models are trained by SGD optimizer with 0.9 momentum and the training parameters shown in Table 8; in linear-probe transfer learning (Table 4a), we apply L-BFGS optimizer to train a classifier for frozen features.

8. Additional results

8.1. Deep models

We additionally apply deep models of DenseNet [46], ResNeXt [49] and MobileNet-v2 [48] to ImageNet training, and report performance results in Table 9 which are measured in the same manner as Table 2; while DenseNet and ResNeXt are trained in the procedure of Table 8, the training parameters for MobileNet-v2 are slightly modified in weight decay of 0.00004 and learning rate of 0.045 which is exponentially decayed. So pre-trained models are then applied to the tasks regarding model confidence (Section 3.2.1, Figure 4) and transfer learning (Section 3.2.2, Table 4); the results are shown in Tables 12,13&14. Similarly to Section 3.2.1&3.2.2, we can observe that (1) the optimized κ^* works as a lower bound, (2) the middle $\kappa = 2\kappa^*$ roughly maximizes performance on ImageNet, (3) the smaller $\kappa \approx \kappa^*$ renders high robustness against less-confident samples, and (4) the larger κ exhibits favorable generalization performance on transfer learning.

8.2. "Cosify"

Table 10 shows performance of ResNet-50 *cosified* with our LS-optimized κ^*. In the *cosification* (via fine-tuning), the LS method robustly produces the same κ^*. Besides, the performances of transfer learning by all the *cosified* models are detailed in Table 15. The same discussion/analysis as in Section 3.2.2 can hold for these results.

8.3. Computation time

Our LS method is composed of two processes, computing reconstruction $\mathcal{E}_\kappa^{LS}$ in (10) and searching minium over κ

Table 9. Image classification accuracy (%) by various models on ImageNet ($C = 1000$). These results augment Table 2.

Model	DenseNet161 [46]	ResNeXt-50 [49]	MobileNet-v2 [48]
d	2208	2048	1280
softmax	78.97	78.20	68.69
LS (11)	78.65	78.46	66.04
(κ^*)	(12.50)	(12.57)	(14.49)
Fix $\kappa = 10$	78.42	78.34	65.23
$\kappa = 20$	78.81	78.29	67.63
$\kappa = 30$	78.89	78.20	68.94
$\kappa = 40$	78.76	78.02	69.41
$\kappa = 50$	78.61	77.87	69.49
$\kappa = 60$	78.08	77.38	69.32
$\kappa = 2\kappa^*$	78.75	78.28	68.96

Table 10. Performance of *cosified* ResNet50 with LS-optimized κ^*. These scores are measured in the ways of Table 2 and Figure 4.

finetune blocks				LS	ImageNet	Specialization			
						AP for MISS		AP for OOD	
1	2	3	4	κ^*	Acc.	Z_{max}	$\|\boldsymbol{x}\|_2$	Z_{max}	$\|\boldsymbol{x}\|_2$
from scratch				12.83	77.27	0.9400	0.9011	0.9016	0.8428
✓	✓	✓	✓	12.79	77.19	0.9413	0.9062	0.9030	0.8474
-	✓	✓	✓	12.77	77.09	0.9420	0.9063	0.9040	0.8555
-	-	✓	✓	12.77	77.10	0.9415	0.9066	0.9055	0.8521
-	-	-	✓	12.78	76.84	0.9410	0.9060	0.9008	0.8533

Table 11. Computation time (msec) to process a mini-batch and optimize κ by LS.

	whole	Optimizing κ	
	mini-batch	$\mathcal{E}_\kappa^{LS}$	$\arg\min_\kappa$
Food101	304	0.13	0.015
Cifar10	33.8	0.118	0.013

candidates $\arg\min_{\kappa\in\mathcal{K}}$ in (14). Table 11 shows computation time of those processes in comparison to a mini-batch computation, demonstrating that they are performed in a negligible computation cost.

References

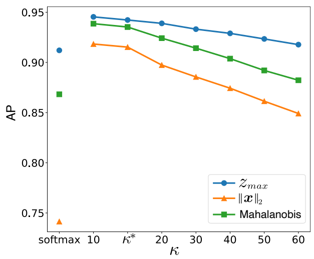
[46] Gao Huang, Zhuang Liu, Laurens van der Maaten, and Kilian Q. Weinberger. Densely connected convolutional networks. In *CVPR*, pages 2261–2269, 2017. 2, 3, 4

[47] Takumi Kobayashi. Three viewpoints toward exemplar svm. In *CVPR*, pages 2765–2773, 2015. 1

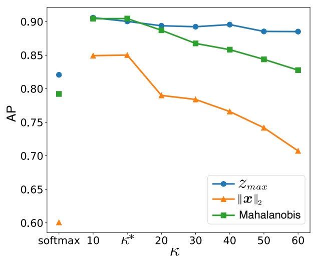
[48] Mark Sandler, Andrew Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen. Mobilenetv2: Inverted residuals and linear bottlenecks. In *CVPR*, pages 4510–4520, 2018. 2, 3, 5

[49] Saining Xie, Ross Girshick, Piotr Dollár, Zhuowen Tu, and Kaiming He. Aggregated residual transformations for deep neural networks. In *CVPR*, pages 5987–5995, 2017. 2, 3, 4

Table 12. Performance results of DenseNet161 [46] pretrained on ImageNet, for detecting miss-classified (MISS) and out-of-distribution (OOD) samples in (a,b) (see Section 3.2.1) and transfer learning in (c) (see Section 3.2.2).



(a) MISS detection

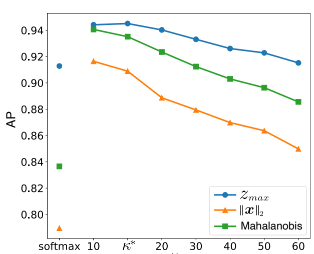


(b) OOD detection

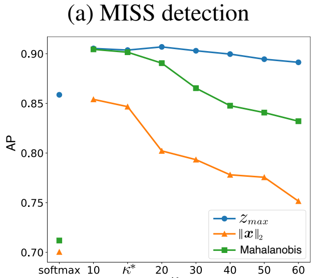
(c) Classification accuracy (%) by transfer learning

Dataset	CUB-200	Food-101	Car-196	Aircraft-100	SUN-397	DTD	Flower-102
Linear probe							
softmax	74.91	74.62	58.97	50.86	61.97	74.36	90.75
Fixed κ = 10	33.98	40.80	16.52	12.57	41.42	58.24	39.70
20	57.01	63.77	39.58	40.02	54.90	67.98	77.56
30	70.09	71.14	55.29	50.71	59.22	73.24	88.00
40	75.68	74.32	61.87	55.69	61.23	74.36	91.97
50	76.42	75.12	64.30	58.54	61.58	74.95	92.13
60	78.93	75.62	66.15	59.59	61.58	74.47	93.48
Fine-tuning							
softmax	80.87	87.52	86.62	77.40	63.08	75.48	95.84
Fixed κ = 10	79.76	86.42	85.72	78.60	61.88	74.63	91.99
20	81.97	87.67	86.72	80.19	63.34	76.60	93.92
30	83.13	87.74	87.71	80.88	64.44	78.09	95.56
40	84.13	87.74	88.29	81.77	64.52	76.91	96.47
50	83.78	87.82	88.51	80.31	65.03	78.03	96.67
60	83.30	87.86	88.72	81.26	64.96	78.09	96.96

Table 13. Performance results of ResNeXt-50 [49] pretrained on ImageNet, for detecting miss-classified (MISS) and out-of-distribution (OOD) samples in (a,b) (see Section 3.2.1) and transfer learning in (c) (see Section 3.2.2).



(a) MISS detection

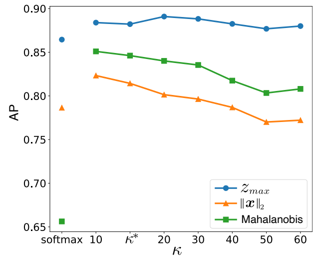


(b) OOD detection

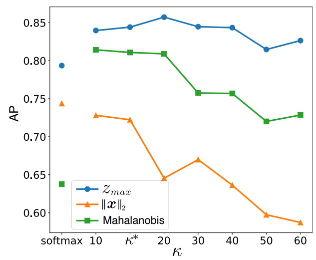
(c) Classification accuracy (%) by transfer learning

Dataset	CUB-200	Food-101	Car-196	Aircraft-100	SUN-397	DTD	Flower-102
Linear probe							
softmax	69.61	69.40	49.43	42.93	59.78	72.98	88.26
Fixed κ = 10	44.41	45.73	20.63	16.89	43.26	57.29	48.93
20	63.51	64.79	43.95	41.40	54.57	69.52	79.80
30	69.16	69.05	53.05	48.84	58.36	72.71	84.47
40	68.43	69.49	53.86	55.09	58.87	72.55	88.78
50	75.20	73.40	61.80	54.07	60.93	74.84	91.07
60	75.98	74.34	63.55	58.48	61.06	73.83	91.25
Fine-tuning							
softmax	80.64	86.62	86.23	77.04	62.87	74.84	94.91
Fixed κ = 10	79.69	86.58	85.43	77.13	63.04	75.27	91.21
20	81.26	87.22	86.70	77.67	63.68	75.64	93.11
30	81.18	87.20	87.36	78.42	63.96	77.23	94.67
40	77.24	86.98	86.77	79.41	64.29	76.86	95.32
50	82.68	87.17	87.52	79.32	64.75	78.35	95.97
60	82.21	87.30	87.47	79.14	64.34	77.87	96.52

Table 14. Performance results of MobileNet-v2 [48] pretrained on ImageNet, for detecting miss-classified (MISS) and out-of-distribution (OOD) samples in (a,b) (see Section 3.2.1) and transfer learning in (c) (see Section 3.2.2).



(a) MISS detection



(b) OOD detection

(c) Classification accuracy (%) by transfer learning

Dataset	CUB-200	Food-101	Car-196	Aircraft-100	SUN-397	DTD	Flower-102
Linear probe							
softmax	68.16	67.45	48.85	48.54	55.60	70.16	89.67
Fixed $\kappa = 10$	37.94	43.68	18.54	20.43	39.54	56.17	51.86
20	57.01	59.81	32.86	38.07	49.84	65.80	76.35
30	66.55	64.34	44.34	45.96	53.41	67.55	85.41
40	68.76	67.34	49.67	48.42	54.87	68.40	87.95
50	70.40	68.89	53.26	51.25	55.69	67.61	89.95
60	70.28	69.37	54.50	52.42	55.52	69.95	90.10
Fine-tuning							
softmax	74.59	83.27	78.95	71.10	58.89	71.81	93.60
Fixed $\kappa = 10$	71.93	81.60	75.07	64.24	56.64	69.84	88.15
20	74.24	82.88	78.39	67.99	58.09	72.18	91.68
30	75.43	83.52	79.76	69.48	59.38	73.30	93.01
40	75.97	83.51	79.75	69.45	59.55	73.14	94.13
50	76.64	83.60	80.36	69.90	59.66	72.61	94.20
60	76.79	83.26	79.89	69.12	59.81	73.46	94.54

Table 15. Classification accuracy (%) by transfer learning of *cosified* ResNet-50 with various κ.

	finetune blocks 1	2	3	4	CUB-200	Food-101	Car-196	Aircraft-100	SUN-397	DTD	Flower-102
Linear probe	$\kappa = 10$; from scratch				47.13	48.23	23.44	22.29	44.69	59.15	55.47
	✓	✓	✓	✓	51.28	54.77	30.17	27.90	47.37	62.55	65.31
	-	✓	✓	✓	50.79	54.72	30.52	29.46	47.01	62.13	63.82
	-	-	✓	✓	51.04	54.78	29.03	29.04	47.16	61.70	63.65
	-	-	-	✓	49.91	54.99	29.85	28.32	46.90	61.54	64.87
	$\kappa = 60$; from scratch				75.63	74.04	61.86	57.16	61.14	74.10	92.19
	✓	✓	✓	✓	71.52	71.09	52.88	48.45	60.11	75.64	89.80
	-	✓	✓	✓	71.49	71.07	52.38	48.57	60.25	75.27	89.77
	-	-	✓	✓	71.54	70.58	52.46	47.94	59.93	74.15	89.14
	-	-	-	✓	70.88	70.36	51.04	47.67	59.82	74.04	88.40
	LS κ^*; from scratch				52.62	57.04	27.50	26.49	50.22	66.91	63.05
	✓	✓	✓	✓	57.78	61.51	37.78	34.26	52.60	67.77	76.44
	-	✓	✓	✓	56.94	61.87	37.15	33.69	52.61	66.01	75.62
	-	-	✓	✓	57.37	61.68	35.67	32.46	52.08	67.34	76.09
	-	-	-	✓	56.71	61.22	37.26	32.85	51.91	67.77	75.36
Fine-tuning	$\kappa = 10$; from scratch				79.76	86.28	85.32	76.71	62.22	73.56	90.57
	✓	✓	✓	✓	79.43	86.47	85.78	78.72	62.93	73.56	90.65
	-	✓	✓	✓	80.21	86.20	85.60	77.82	62.61	73.94	91.34
	-	-	✓	✓	79.62	86.65	85.90	79.11	62.76	74.15	91.11
	-	-	-	✓	79.54	86.39	85.68	79.08	63.14	74.57	90.91
	$\kappa = 60$; from scratch				81.18	86.83	86.66	78.12	63.80	76.49	96.36
	✓	✓	✓	✓	82.13	86.69	87.18	78.63	63.15	76.86	95.16
	-	✓	✓	✓	82.59	86.82	87.07	78.96	63.18	76.70	94.93
	-	-	✓	✓	82.66	86.79	86.88	79.23	63.37	76.86	95.01
	-	-	-	✓	82.59	86.74	87.00	79.26	63.28	76.91	95.20
	LS κ^*; from scratch				78.86	86.14	84.22	75.33	62.78	74.52	90.57
	✓	✓	✓	✓	79.71	86.56	84.71	77.79	62.93	75.59	92.08
	-	✓	✓	✓	80.23	86.45	85.13	77.67	63.07	74.68	92.47
	-	-	✓	✓	79.81	86.23	85.06	77.49	62.67	74.89	92.20
	-	-	-	✓	79.92	86.36	85.72	78.84	62.82	74.47	91.86