Learning for Single-Shot Confidence Calibration in Deep Neural Networks through Stochastic Inferences

Supplementary Document

A. Stochastic depth as approximate Bayesian inference

ResNet [2] proposes to add skip connections to the network. If x_i denotes the output of the i^{th} layer and $f_i(x)$ represents a typical convolutional transformation, we can obtain the forward propagation

$$\mathbf{x}_i = f_i(\mathbf{x}_{i-1}) + \mathbf{x}_{i-1} \tag{1}$$

and $f_i(\mathbf{x})$ is commonly defined by

$$f_i(\mathbf{x}) = W_i \cdot \sigma(B(W_i{}' \cdot \sigma(B(\mathbf{x})))) \tag{2}$$

where W_i and $W_i{}'$ are weight matrices, $(\cdot)$ denotes convolution, and B and σ indicates batch normalization and ReLU function, respectively.

ResNet with stochastic depth [3] randomly drops a subset of layers and bypass them with short-cut connection. Let $e_i \in \{0, 1\}$ denotes a Bernoulli random variable which indicates whether the i^{th} residual block is active or not. The forward propagation is extended from Equation 1 to

$$\mathbf{x}_i = e_i f_i(\mathbf{x}_{i-1}) + \mathbf{x}_{i-1}. \tag{3}$$

Now we can transform stochasticity from layers to the parameter space as follows:

$$\begin{aligned}
e_i f_i(\mathbf{x}_{i-1}) + \mathbf{x}_{i-1} &= e_i\big(W_i \cdot \sigma(B(W_i{}' \cdot \sigma(B'(\mathbf{x}_{i-1}))))\big) + \mathbf{x}_{i-1} && (4)\\
&= e_i^4\big(W_i \cdot \sigma(B(W_i{}' \cdot \sigma(B'(\mathbf{x}_{i-1}))))\big) + \mathbf{x}_{i-1} && (5)\\
&= e_i^4\left(W_i \cdot \sigma(\gamma_i(\frac{W_i{}' \cdot \sigma(B'(\mathbf{x}_{i-1})) - \mu_i}{\delta_i}) + \beta_i)\right) + \mathbf{x}_{i-1} && (6)\\
&= e_i W_i \cdot \sigma(e_i \begin{bmatrix}\gamma_i\\ \beta_i\end{bmatrix}^T \begin{bmatrix}\frac{1}{\delta_i}(e_i W_i{}' \cdot \sigma(e_i\gamma_i'(\overline{\mathbf{x}_{i-1}} + \beta_i')) - e_i\mu_i)\\ 1\end{bmatrix}) + \mathbf{x}_{i-1} && (7)\\
&= \widetilde{W}_i \cdot \sigma(\begin{bmatrix}\widetilde{\gamma}_i\\ \widetilde{\beta}_i\end{bmatrix}^T \begin{bmatrix}\frac{1}{\delta_i}(\widetilde{W}_i{}' \cdot \sigma(\widetilde{\gamma}_i'(\overline{\mathbf{x}_{i-1}} + \widetilde{\beta}_i')) - e_i\mu_i)\\ 1\end{bmatrix}) + \mathbf{x}_{i-1} && (8)\\
&= f_{\widetilde{W}_i, \widetilde{W}_i{}', \widetilde{\gamma}_i, \widetilde{\gamma}_i{}', \widetilde{\beta}_i, \widetilde{\beta}_i{}'}(\mathbf{x}_{i-1}) = f_{\widetilde{\omega}_i}(\mathbf{x}_{i-1}) && (9)
\end{aligned}$$

$e_i = e_i^4$ since it is a Bernoulli random variable. All stochastic parameters $\widetilde{\omega}_i = [\widetilde{W}_i, \widetilde{W}_i{}', \widetilde{\gamma}_i, \widetilde{\gamma}_i{}', \widetilde{\beta}_i, \widetilde{\beta}_i{}']$ in this block drop at once or not.

B. Approximation of KL-Divergence

Let $\omega \in \mathbb{R}^D$, $p(\omega) = \mathcal{N}(0, \mathbf{I})$ and $q_\theta(\omega) = \sum_{i=1}^{2} e_i \mathcal{N}(\theta_i, \sigma^2 \mathbf{I})$ with a probability vector $\mathbf{e} = (e_1, e_2)$ where $e_i \in [0, 1]$ and $\sum_{i=1}^{2} e_i = 1$. In our work, θ_1 denotes the deterministic model parameter $[W_i, W_i', \gamma_i, \gamma_i', \beta_i, \beta_i']$ and $\theta_2 = 0$. The KL-Divergence between $q_\theta(\omega)$ and $p(\omega)$ is

$$\mathrm{KL}(q_\theta(\omega)||p(\omega)) = \int q_\theta(\omega) \log \frac{q_\theta(\omega)}{p(\omega)} d\omega \tag{10}$$

$$= \int q_\theta(\omega) \log q_\theta(\omega) d\omega - \int q_\theta(\omega) \log p(\omega) d\omega. \tag{11}$$

We can re-parameterize the first entropy term with $\omega = \theta_i + \sigma\epsilon_i$ where $\epsilon_i \sim \mathcal{N}(0, \mathbf{I})$.

$$\int q_\theta(\omega) \log q_\theta(\omega) d\omega = \sum_{i=1}^{2} e_i \int \mathcal{N}(\omega; \theta_i, \sigma_i^2 \mathbf{I}) \log q_\theta(\omega) d\omega \tag{12}$$

$$= \sum_{i=1}^{2} e_i \int \mathcal{N}(\epsilon_i; 0, \mathbf{I}) \log q_\theta(\theta_i + \sigma\epsilon_i) d\omega \tag{13}$$

Using $q_\theta(\theta_i + \sigma\epsilon_i) \approx e_i (2\pi)^{-D/2} \sigma^{-1} \exp(-\frac{1}{2}\epsilon_i^T \epsilon_i)$ for large enough D,

$$\int q_\theta(\omega) \log q_\theta(\omega) d\omega \approx \sum_{i=1}^{2} e_i \int \mathcal{N}(\epsilon_i; 0, \mathbf{I}) \log \big(e_i (2\pi)^{-D/2} \sigma^{-1} \exp(-\frac{1}{2}\epsilon_i^T \epsilon_i)\big) d\omega \tag{14}$$

$$= \sum_{i=1}^{2} \frac{e_i}{2} \Big(\log e_i - \log \sigma + \int \mathcal{N}(\epsilon_i; 0, \mathbf{I}) \epsilon_i^T \epsilon_i d\epsilon_i - D \log 2\pi \Big) \tag{15}$$

$$\approx \sum_{i=1}^{2} \frac{e_i}{2} \big(- \log \sigma - D(1 + \log 2\pi) \big) - \frac{1}{2} H(e). \tag{16}$$

For the second term of the Equation 11,

$$\int q_\theta(\omega) \log p(\omega) d\omega = \sum_{i=1}^{2} e_i \int \mathcal{N}(\omega; \theta_i, \sigma_i^2 \mathbf{I}) \log \mathcal{N}(\omega; 0, \mathbf{I}) d\omega \tag{17}$$

$$= -\frac{1}{2} \sum_{i=1}^{2} e_i \big(\theta_i^T \theta_i + D\sigma \big). \tag{18}$$

Then we can approximate

$$\mathrm{KL}(q_\theta(\omega)||p(\omega)) \approx \sum_{i=1}^{2} \frac{e_i}{2} \big(\theta_i^T \theta_i + D\sigma - \log \sigma - D(1 + \log 2\pi) \big) - \frac{1}{2} H(e). \tag{19}$$

For a more general proof, see [1].

C. Full Experimental Results

We present the full results of Table 1 in the main paper including all scores from individual CI models with different β's. Table A and B show the results on Tiny ImageNet and CIFAR-100, respectively.

Table A. Classification accuracy and calibration scores of the models trained with various architectures on Tiny ImageNet supplementing Table 1 in the main paper. This table includes results from all models trained with CI loss with different β's. In addition to the architectures presented in Table 1, we also include results from ResNet-18.

Dataset	Architecture	Method	Accuracy[%]	ECE	MCE	NLL	Brier Score
Tiny ImageNet	ResNet-18	Baseline	46.38	0.029	0.086	2.227	0.674
		CI[$\beta = 10^{-4}$]	46.48	0.022	0.073	2.216	0.672
		CI[$\beta = 10^{-3}$]	47.20	0.022	0.060	2.198	0.666
		CI[$\beta = 0.01$]	47.03	**0.021**	0.157	2.193	0.667
		CI[$\beta = 0.1$]	47.58	0.055	0.111	2.212	0.666
		CI[$\beta = 1$]	47.92	0.241	0.380	2.664	0.742
		VWCI	**48.57**	0.026	**0.054**	**2.129**	**0.651**
	ResNet-34	Baseline	50.82	0.067	0.147	2.050	0.628
		CI[$\beta = 10^{-4}$]	48.89	0.132	0.241	2.257	0.668
		CI[$\beta = 10^{-3}$]	50.17	0.127	0.227	2.225	0.653
		CI[$\beta = 0.01$]	49.16	0.119	0.219	2.223	0.663
		CI[$\beta = 0.1$]	51.45	0.035	0.171	2.030	0.620
		CI[$\beta = 1$]	50.77	0.255	0.426	2.614	0.722
		VWCI	**52.80**	**0.027**	**0.076**	**1.949**	**0.605**
	VGG-16	Baseline	46.58	0.346	0.595	4.220	0.844
		CI[$\beta = 10^{-4}$]	47.26	0.325	0.533	3.878	0.830
		CI[$\beta = 10^{-3}$]	47.39	0.296	0.536	3.542	0.795
		CI[$\beta = 0.01$]	47.11	0.259	0.461	3.046	0.763
		CI[$\beta = 0.1$]	46.94	0.122	0.327	2.812	0.701
		CI[$\beta = 1$]	45.40	0.130	0.320	2.843	0.717
		VWCI	**48.03**	**0.053**	**0.142**	**2.373**	**0.659**
	WideResNet-16-8	Baseline	55.92	0.132	0.237	1.974	0.593
		CI[$\beta = 10^{-4}$]	55.29	0.126	0.208	1.987	0.598
		CI[$\beta = 10^{-3}$]	55.53	0.120	0.237	1.949	0.592
		CI[$\beta = 0.01$]	56.12	0.116	0.238	1.949	0.590
		CI[$\beta = 0.1$]	56.38	0.050	0.456	**1.851**	0.572
		CI[$\beta = 1$]	55.66	0.161	0.301	2.163	0.619
		VWCI	**56.66**	**0.046**	**0.136**	1.866	**0.569**
	DenseNet-40-12	Baseline	42.50	**0.020**	0.154	2.423	0.716
		CI[$\beta = 10^{-4}$]	41.20	0.030	0.156	2.489	0.726
		CI[$\beta = 10^{-3}$]	41.21	0.036	0.122	2.514	0.735
		CI[$\beta = 0.01$]	40.61	0.025	0.097	2.550	0.739
		CI[$\beta = 0.1$]	40.67	0.037	0.094	2.501	0.732
		CI[$\beta = 1$]	37.23	0.169	0.291	2.975	0.810
		VWCI	**43.25**	0.025	**0.089**	**2.410**	**0.712**

Table B. Classification accuracy and calibration scores of models trained with various methods on CIFAR-100 supplementing Table 1 in the main paper. This table includes results from all models trained with CI loss with different β's. In addition to the architectures presented in Table 1, we also include results from ResNet-18.

Dataset	Architecture	Method	Accuracy[%]	ECE	MCE	NLL	Brier Score
CIFAR-100	ResNet-18	Baseline	75.61	0.097	0.233	1.024	0.359
		CI[$\beta = 10^{-4}$]	75.03	0.104	0.901	1.055	0.369
		CI[$\beta = 10^{-3}$]	75.51	0.087	0.219	0.986	0.357
		CI[$\beta = 0.01$]	74.95	0.069	0.183	0.998	0.358
		CI[$\beta = 0.1$]	75.94	0.065	0.961	1.018	0.349
		CI[$\beta = 1$]	75.61	0.340	0.449	1.492	0.475
		VWCI	**76.09**	**0.045**	**0.128**	**0.976**	**0.342**
	ResNet-34	Baseline	77.19	0.109	0.304	1.020	0.345
		CI[$\beta = 10^{-4}$]	77.38	0.105	0.259	1.000	0.341
		CI[$\beta = 10^{-3}$]	76.98	0.101	0.261	0.999	0.344
		CI[$\beta = 0.01$]	77.23	0.074	0.206	0.921	0.331
		CI[$\beta = 0.1$]	77.66	**0.029**	**0.087**	0.953	0.321
		CI[$\beta = 1$]	78.54	0.362	0.442	1.448	0.461
		VWCI	**78.64**	0.034	0.089	**0.908**	**0.310**
	VGG-16	Baseline	73.78	0.187	0.486	1.667	0.437
		CI[$\beta = 10^{-4}$]	73.19	0.189	0.860	1.679	0.446
		CI[$\beta = 10^{-3}$]	73.70	0.183	0.437	1.585	0.434
		CI[$\beta = 0.01$]	73.78	0.163	0.425	1.375	0.420
		CI[$\beta = 0.1$]	73.68	**0.083**	**0.285**	1.289	0.396
		CI[$\beta = 1$]	73.62	0.291	0.399	1.676	0.487
		VWCI	**73.87**	0.098	0.309	**1.277**	**0.391**
	WideResNet-16-8	Baseline	77.52	0.103	0.278	0.984	0.336
		CI[$\beta = 10^{-4}$]	77.04	0.109	0.280	1.011	0.345
		CI[$\beta = 10^{-3}$]	77.46	0.104	0.272	0.974	0.339
		CI[$\beta = 0.01$]	77.53	0.074	0.211	0.931	0.327
		CI[$\beta = 0.1$]	77.23	0.085	0.239	1.015	0.336
		CI[$\beta = 1$]	77.48	0.295	0.485	1.378	0.434
		VWCI	**77.74**	**0.038**	**0.101**	**0.891**	**0.314**
	DenseNet-40-12	Baseline	65.91	0.074	0.134	1.238	0.463
		CI[$\beta = 10^{-4}$]	66.20	0.064	0.141	1.236	0.463
		CI[$\beta = 10^{-3}$]	63.61	0.086	0.177	1.360	0.496
		CI[$\beta = 0.01$]	65.13	0.052	0.127	1.249	0.471
		CI[$\beta = 0.1$]	65.86	**0.019**	**0.053**	1.206	0.456
		CI[$\beta = 1$]	62.82	0.127	0.193	1.510	0.523
		VWCI	**67.45**	0.026	0.094	**1.161**	**0.439**

References

[1] Yarin Gal and Zoubin Ghahramani. Dropout as a bayesian approximation: Representing model uncertainty in deep learning. In *ICML*, 2016. 2

[2] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, 2016. 1

[3] Gao Huang, Yu Sun, Zhuang Liu, Daniel Sedra, and Kilian Q Weinberger. Deep networks with stochastic depth. In *ECCV*, 2016. 1