C-LoRA: Contextual Low-Rank Adaptation for Uncertainty Estimation in Large Language Models

Amir Hossein Rahmati, Sanket Jantre, et al. (NeurIPS 2025)

June 4, 2026

YunSeop Shin, Seoul National University, Statistics, IDEA LAB

Notation

- $x^l \in \mathbb{R}^k$, $h^l \in \mathbb{R}^d$: layer ($l = 1, \cdots, L$) input / output (authors set $k = d$).
- $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^N$: fine-tuning dataset.
- $W_0^l \in \mathbb{R}^{d \times d}$: *frozen* pre-trained weight; ΔW^l: learned update.
- $B^l \in \mathbb{R}^{d \times r}$, $A^l \in \mathbb{R}^{r \times d}$: LoRA factors, rank $r \ll d$.
- $E^l \in \mathbb{R}^{r \times r}$: lightweight middle matrix.
- E_x^l: **data-dependent** version of E^l, with

$$q_\phi(E_x^l \mid x) = \mathcal{N}\big(\mu_\phi^l(x), \{\Omega_\phi^l(x)\}^2\big).$$

- $\theta = \{B^l, A^l\}_{l=1}^L$: *deterministic* adapter parameters.
- $\phi = \{\phi^l\}_{l=1}^L$: *contextual module* parameters (small per-layer NNs g_ϕ^l).
- $z^l = A^l x^{l-1} \in \mathbb{R}^r$: low-dim input fed to the contextual module.
- $p(y \mid x, \theta)$: likelihood.

Motivation

- LoRA fine-tunes LLMs, but it tends to be **overconfident** and hallucinate.
- Bayesian LoRA methods add uncertainty, but mostly target model uncertainty:
  - Laplace-LoRA: post-hoc Laplace approximation.
  - BLoB: Variational inference over the LoRA A matrix.
- Key limitation: their weight distribution is input-independent, so the uncertainty estimate does *not* adapt to each input.

Idea of C-LoRA

Make the LoRA weight distribution **depend on the input** x, so that uncertainty is estimated per sample.

Preliminary

- LoRA freezes W_0^l and learns a **low-rank** update $\Delta W^l = B^l A^l$:

$$h^l = (W_0^l + \Delta W^l)x^{l-1} = (W_0^l + B^l A^l)x^{l-1}. \tag{1}$$

- Trainable parameters: $r \times (d + k)$ instead of $d \times k$, with $r \ll d$.
- Full posterior over LLM (or even all LoRA) weights is intractable.

Preliminary

- Bayesian Low-Rank Adaptation by Backpropagation (BLoB) keeps B deterministic and A random with a prior $p(A)$ and proxy distribution $q(A) = \mathcal{N}(A \mid \mu_A, \Omega_A^2)$.
- Trains μ_A, Ω_A by maximizing the ELBO:

$$\mathcal{L}' = \underbrace{\mathbb{E}_q\big[\log p(\mathcal{D} \mid A, B)\big]}_{\text{expected log-likelihood}} - \underbrace{\mathrm{KL}\big(q(A) \,\|\, p(A)\big)}_{\text{regularizer}}. \tag{2}$$

- $A \in \mathbb{R}^{r \times d}$, so the number of stochastic parameters grows in d.

Method

- Insert a small middle matrix $E^l \in \mathbb{R}^{r \times r}$ between B^l and A^l:

$$h^l = (W_0^l + \Delta W^l) x^{l-1} = (W_0^l + B^l E^l A^l)\, x^{l-1}. \tag{3}$$

- Make **only** E^l **stochastic**; learn B^l and A^l *deterministically*.
- Effect on cost: the number of random parameters becomes

$$\underbrace{r \times d}_{\text{BLoB: scales with } d} \longrightarrow \underbrace{r \times r}_{\text{C-LoRA: constant in } d}.$$

- This $r \times r$ stochastic matrix is what makes a data-dependent Bayesian method scalable enough to apply to LLMs.

Model

- Let the middle matrix depend on the input: $\Delta W^l = B^l E_x^l A^l$ such that

$$q_\phi(E_x^l \mid x^{l-1}) = \mathcal{N}\big(\mu_\phi(x^{l-1}), \Omega_\phi^2(x^{l-1})\big). \tag{4}$$

- A network g_ϕ^l predicts the variational parameters of E_x^l for any x^{l-1}.
- Across L layers, $E_x = (E_x^1, \cdots, E_x^l)$ is factorized following:

$$q_\phi(E_x \mid x) = \prod_{l=1}^{L} q_\phi\big(E_x^l \mid x^{l-1}\big). \tag{5}$$

- At layer l: feed $z^l = A^l x^{l-1} \in \mathbb{R}^r$ into a $g_\phi^l \to$ outputs $(\mu_\phi^l, \Omega_\phi^l)$.
- The likelihood is a softmax over the answer tokens:

$$p_\theta(y \mid x, E_x) = \mathrm{softmax}\big(f_\theta(x, E_x)\big)_y, \tag{6}$$

where f_θ is given transformer model.

Learning

- Maximize a ELBO:

$$\mathcal{L} = \sum_{i=1}^{N} \Big[\mathbb{E}_{q_\phi(E_{x_i}|x_i)} \log p_\theta(y_i \mid x_i, E_{x_i}) \; - \; \mathrm{KL}\big(q_\phi(E_{x_i} \mid x_i) \,\|\, p(E_{x_i})\big)\Big]. \tag{7}$$

- **Update θ (B, A):** use only the NLL term (drop KL)

$$\nabla_\theta \mathcal{L}(x, y) = \mathbb{E}_{q_\phi(E_x|x)} \nabla_\theta \log p_\theta(y \mid x, E_x). \tag{8}$$

  Authors approximate this expectation with Monte Carlo.

- **Update ϕ:** appears in *both* NLL and KL; use the reparameterization $E_x^l = \mu_\phi^l + \Omega_\phi^l \odot \mathcal{E}^l$, $\mathcal{E}^l \sim \mathcal{N}(0, I)$, i.e. $E_x = k_\phi(\mathcal{E}, x)$:

$$\nabla_\phi \mathcal{L}(x, y) = \mathbb{E}_{\mathcal{E}\sim\mathcal{N}(0,I)} \left[\nabla_\phi \left(\log p_\theta \left(y \mid x, k_\phi(\mathcal{E}, x)\right) - \log \frac{q_\phi\left(k_\phi(\mathcal{E}, x) \mid x\right)}{p\left(k_\phi(\mathcal{E}, x)\right)} \right) \right]. \tag{9}$$

- Prior $p(E_x)$: a fixed Gaussian.

Inference

- **Deterministic (M=0):** replace E_x by its mean and predict $p\big(y \mid x, \mu_\phi(x)\big)$ — a fast point estimate.
- **Bayesian (M=10):** sample E_x M times and average,

$$p(y_{\text{test}} \mid x_{\text{test}}, \mathcal{D}) = \frac{1}{M} \sum_{m=1}^{M} p\big(y \mid x, E_x^m\big), \quad E_x^m \sim q_\phi(E_x \mid x). \quad (10)$$

Summary

Stochastic matrix E_x^l is tiny ($r \times r$) and input-driven $\Rightarrow$ **scalable, sample-specific** uncertainty.

Experiments

Setup. LLaMA2-7B, 6 commonsense reasoning tasks; metrics: ACC, ECE (calibration), NLL. Baselines: MAP, MC-Dropout, Deep Ensemble, Laplace-LoRA, BLoB.

Metric	Method	Datasets					
		WG-S	ARC-C	ARC-E	WG-M	OBQA	BoolQ
ACC ↑	MAP	69.37±1.04	67.67±1.18	85.20±0.63	74.57±0.73	81.60±0.40	87.68±0.02
	MCD	69.06±1.40	66.66±2.30	85.49±0.74	75.89±0.48	81.46±0.92	87.67±0.08
	Deep Ensemble	68.98±0.97	68.57±2.11	86.24±1.26	77.39±1.08	82.20±0.91	88.07±0.17
	LA	68.18±1.04	64.17±0.97	85.30±0.97	74.15±0.40	77.53±0.80	86.45±0.35
	BLoB (M=0)	69.95±0.95	69.25±0.33	85.79±0.83	75.52±0.36	81.85±0.53	86.63±0.52
	BLoB (M=10)	66.55±0.61	66.66±2.25	84.56±0.20	73.38±0.29	81.44±0.53	86.63±0.50
	C-LoRA (M=0)	67.16±0.27	69.02±1.03	84.74±0.20	72.09±2.70	81.13±1.00	85.84±0.67
	C-LoRA (M=10)	66.21±1.24	67.79±1.27	84.38±0.67	70.48±1.71	78.26±2.61	84.64±0.81
ECE ↓	MAP	29.76±1.08	30.60±1.26	13.49±0.63	23.01±0.44	15.30±0.11	5.93±0.36
	MCD	28.49±1.60	29.60±2.77	12.69±0.60	20.73±0.38	14.34±1.11	5.13±0.25
	Deep Ensemble	28.72±1.46	27.75±1.86	11.87±0.16	18.67±0.29	13.98±1.12	5.24±0.27
	LA	11.41±0.17	30.54±0.70	45.85±2.08	10.80±0.38	35.65±1.14	18.22±0.41
	BLoB (M=0)	21.22±1.67	22.57±1.24	10.13±0.39	12.35±0.86	9.35±1.08	2.90±0.18
	BLoB (M=10)	11.23±1.45	10.77±1.91	4.29±1.08	4.52±0.91	**3.82**±0.96	**1.46**±0.36
	C-LoRA (M=0)	7.16±2.92	12.28±1.01	5.75±1.00	6.07±4.85	5.10±1.36	**1.46**±0.85
	C-LoRA (M=10)	**6.86**±3.99	**8.83**±1.20	**4.27**±1.24	**3.71**±1.30	4.00±0.84	1.62±0.44
NLL ↓	MAP	2.86±0.23	3.07±0.09	1.13±0.10	1.26±0.12	1.04±0.02	0.34±0.00
	MCD	2.50±0.12	2.81±0.05	1.13±0.04	1.16±0.03	1.01±0.07	0.32±0.00
	Deep Ensemble	2.44±0.23	2.20±0.03	0.91±0.05	1.04±0.09	0.87±0.03	0.32±0.00
	LA	**0.62**±0.00	1.17±0.01	0.97±0.05	0.56±0.00	0.98±0.01	0.45±0.00
	BLoB (M=0)	0.90±0.07	1.34±0.05	0.63±0.02	0.61±0.03	0.59±0.02	**0.31**±0.00
	BLoB (M=10)	0.66±0.01	**0.88**±0.03	**0.44**±0.00	**0.54**±0.00	**0.51**±0.01	**0.31**±0.01
	C-LoRA (M=0)	0.64±0.03	0.89±0.09	0.46±0.01	0.58±0.03	0.53±0.01	0.34±0.01
	C-LoRA (M=10)	0.63±0.02	**0.88**±0.00	0.48±0.02	0.57±0.03	0.59±0.05	0.35±0.02