Token-Scaled Logit Distillation for Ternary Weight Generative Language Models

NeurIPS 2023

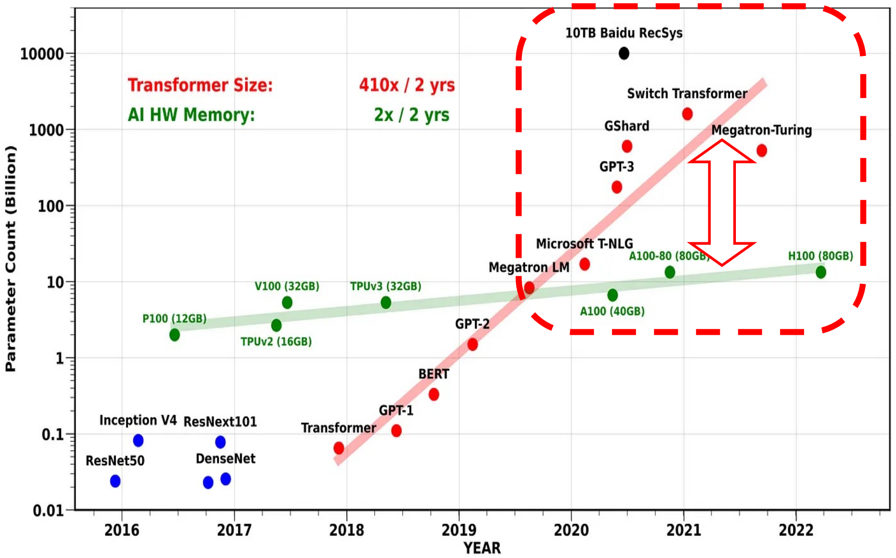
Minsoo Kim, Sihwa Lee, Janghwan Lee, Sukjin Hong, Du-Seong Chang, Wonyong Sung and Jungwook Choi

minsoo2333@hanyang.ac.kr

Challenge of Ternary Large Language Model: Accuracy Loss

- Memory wall in Hyper-scale LLM => **ternary weight quantization**
  - 1) 16x less GPU memory requirement than FP32
  - 2) Multiplication-less MATMUL Implementation

AI and Memory Wall Challenges



https://github.com/amirgholami/ai_and_memory_wall

32-bit

-0.83	0.96	0.04
0.05	-1.10	1.23
-0.02	0.21	-1.23

Ternary Quantization

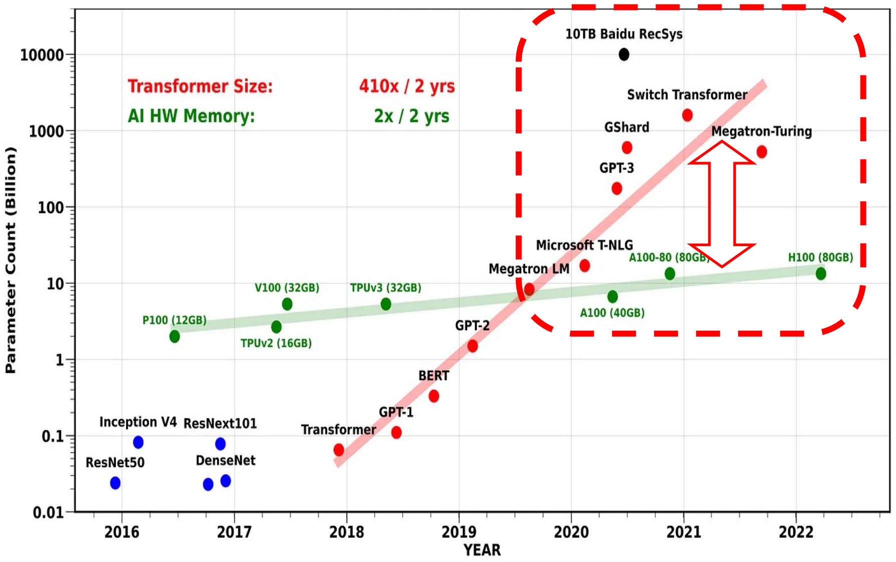
2-bit

-1	1	0
0	-1	1
0	0	-1

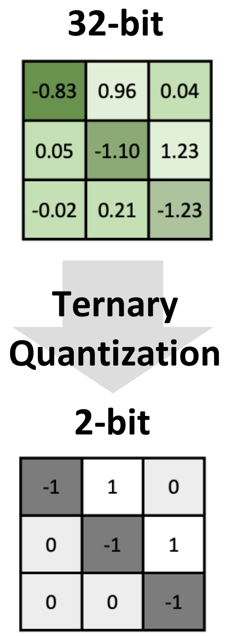
Challenge of Ternary Large Language Model: Accuracy Loss

- Memory wall in Hyper-scale LLM => ternary weight quantization
  - 1) 16x less GPU memory requirement than FP32
  - 2) Multiplication-less MATMUL Implementation
- **Challenge: <u>significant accuracy loss</u> with SOTA LLM compression methods**

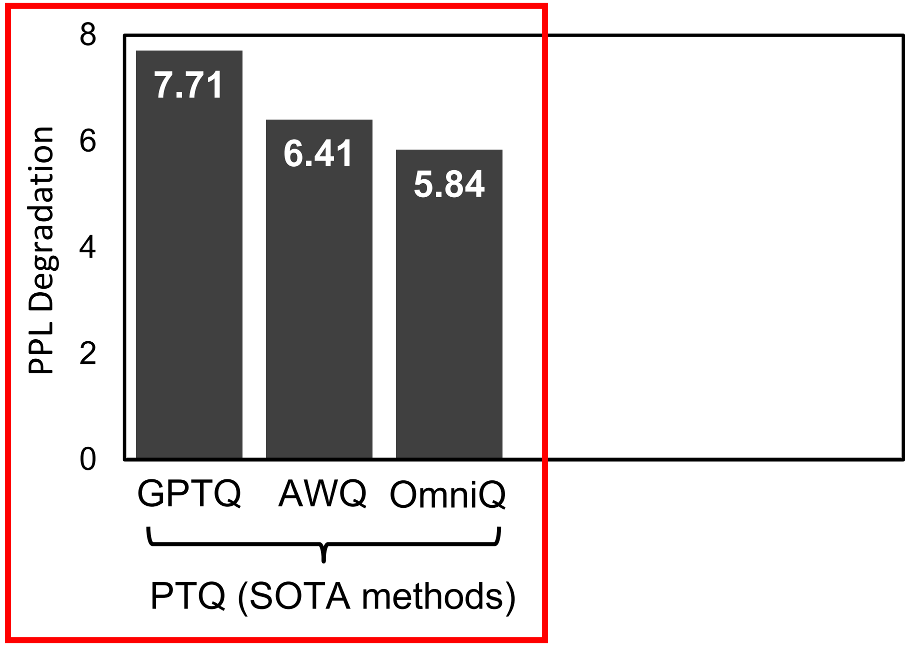
AI and Memory Wall Challenges



https://github.com/amirgholami/ai_and_memory_wall



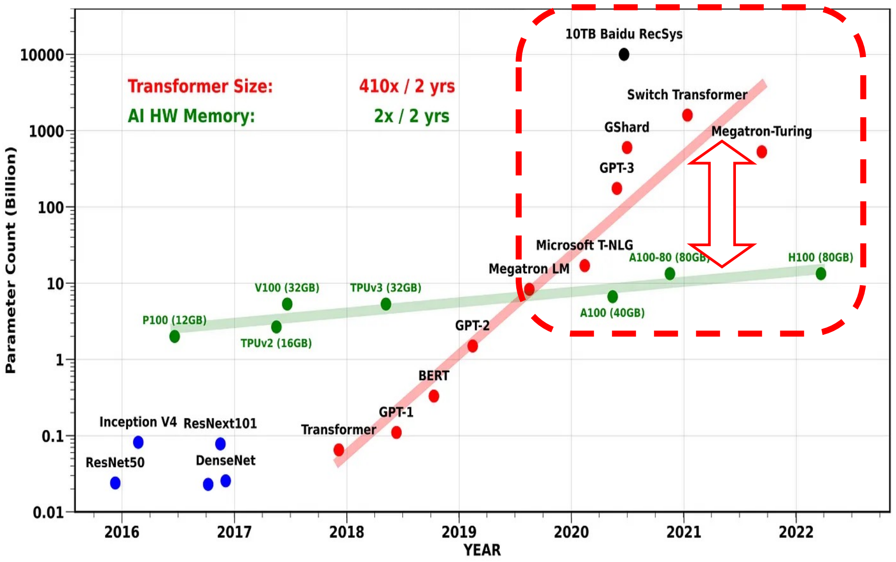
Ternary Weight LLM Performance (OPT-6.7B)



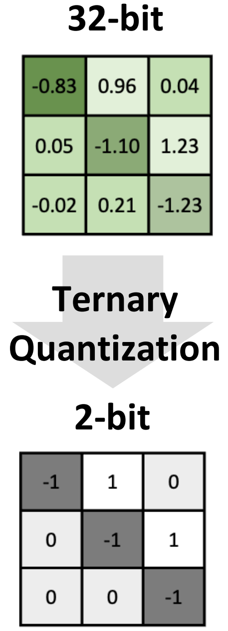
Challenge of Ternary Large Language Model: Accuracy Loss

- Memory wall in Hyper-scale LLM => ternary weight quantization
  - 1) 16x less GPU memory requirement than FP32
  - 2) Multiplication-less MATMUL Implementation
- Challenge: <u>significant accuracy loss</u> with SOTA LLM compression methods

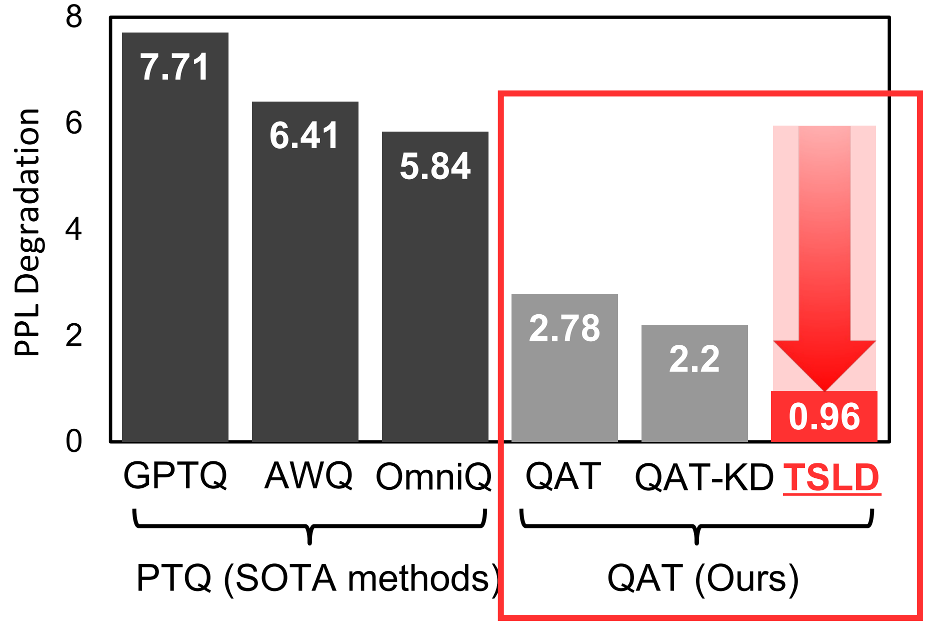
AI and Memory Wall Challenges



https://github.com/amirgholami/ai_and_memory_wall



Ternary Weight LLM Performance (OPT-6.7B)



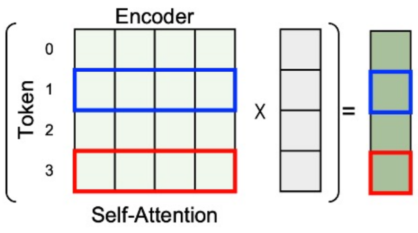
Memory-efficient, superior learning QAT-KD method for GLMs up to 7B

Challenge 1. Cumulative Errors in Causal Attention

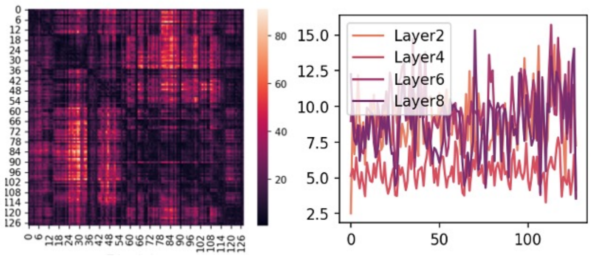
- **Challenge) Cumulative quantization error towards latter tokens in causal attention**

Cumulative Quantization Error

Encoder
(BERT-base)

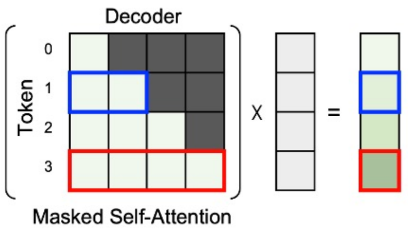


Self-Attention

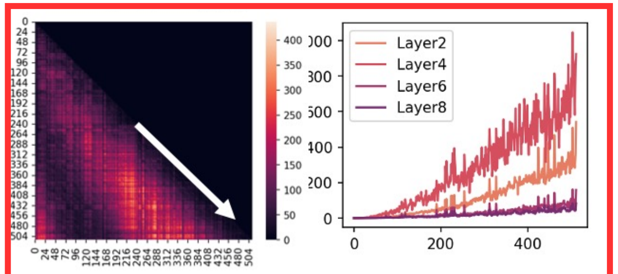


Attention Map Quantization Error

Decoder
(GPT-2)



Causal Attention

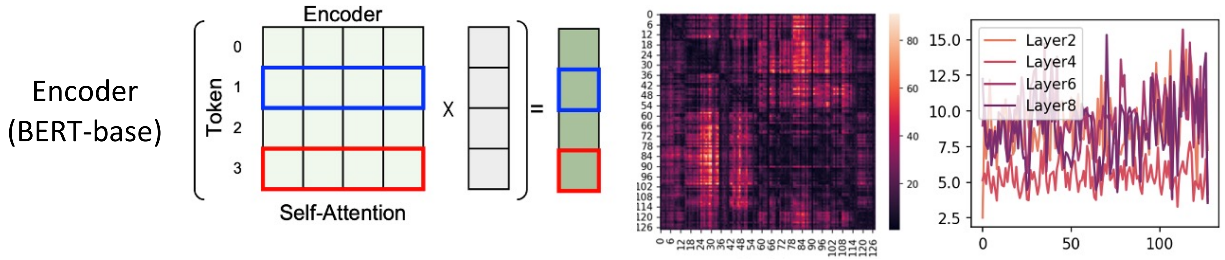


Attention Map Quantization Error

Challenge 1. Cumulative Errors in Causal Attention

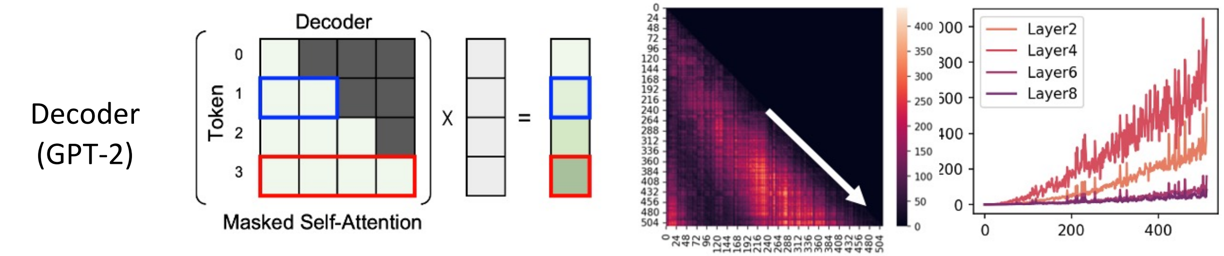
- Challenge) Cumulative quantization error towards latter tokens in causal attention
- **Current KD methods for QAT of Transformer Encoders/Decoders**
  - Layer-to-Layer (L2L) KD: KD on every Transformer layer's output and attention scores
  - Logit KD : cross-entropy loss between final logits from teacher and student model

Cumulative Quantization Error



Self-Attention

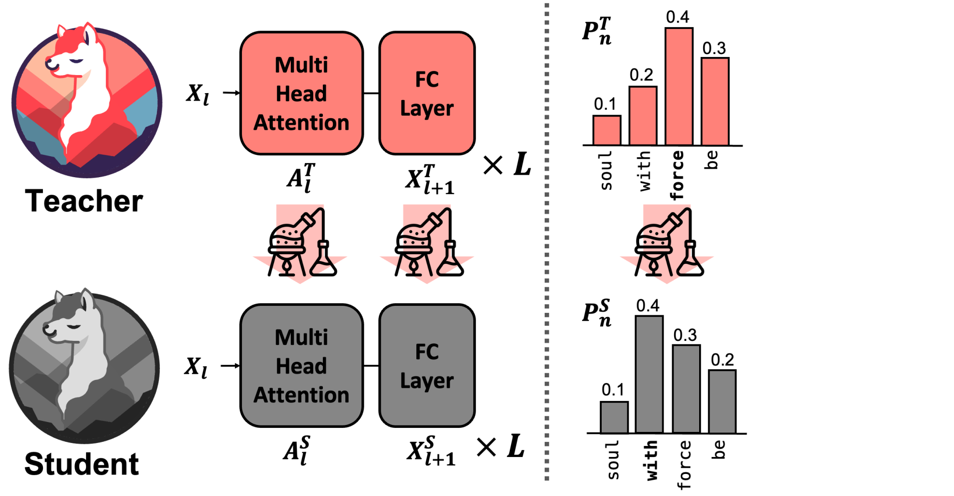
Attention Map Quantization Error



Causal Attention

Attention Map Quantization Error

KD methods for Decoder QAT



L2L KD

Logit KD

$$L_{L2L} = \sum_{l=1}^{L} \mathrm{MSE}(X_l^S, X_l^T) + MSE(A_l^S, A_l^T)$$

$$L_{logit} = -\frac{1}{N}\sum_{n=1}^{N}\sum_{i=1}^{V} P_{n,i}^T \log(P_{n,i}^S)$$

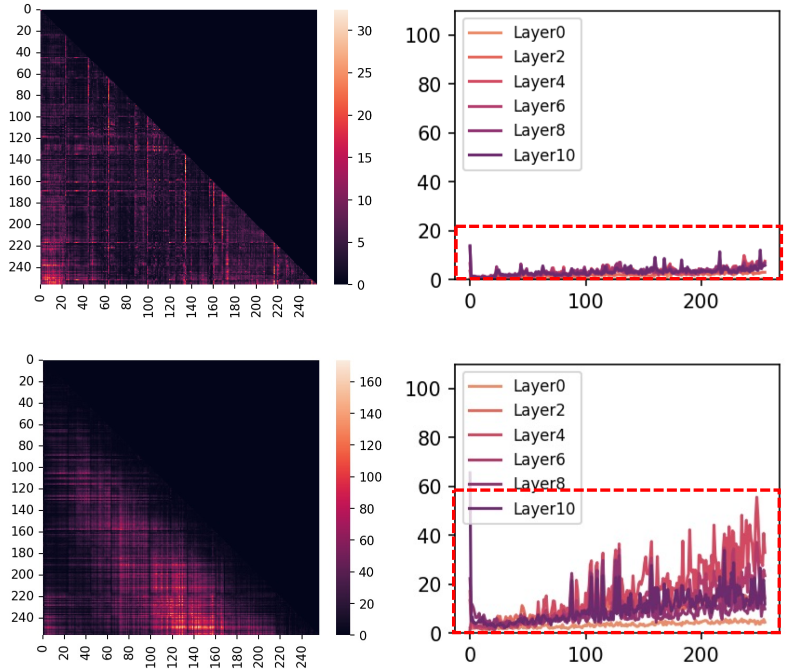
Logit Distillation for Cumulative Quantization Error

- L2L KD fails to align final logit distribution, but Logit KD accurately **reproduce the final logit distribution.**
- **Accurate final logit distance -> Improve accuracy in language modeling task! (lower PPL score)**

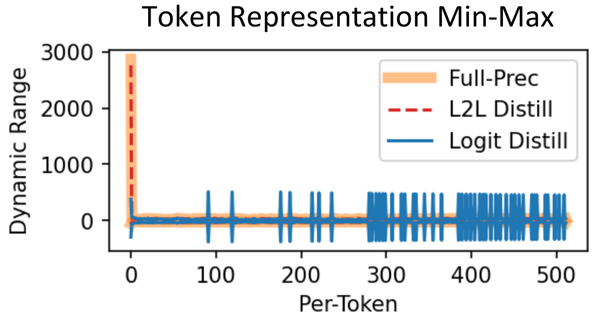
Attention Map Quantization Error

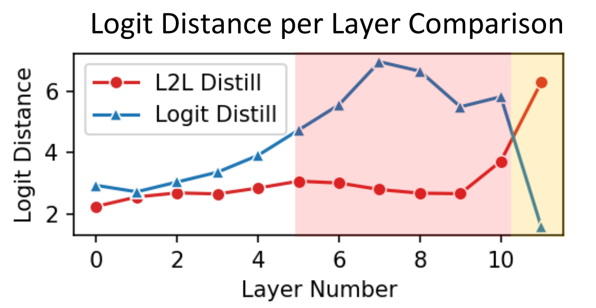
L2L KD

Logit KD



Activation Analysis with Different KDs





Language Modeling PPL Score

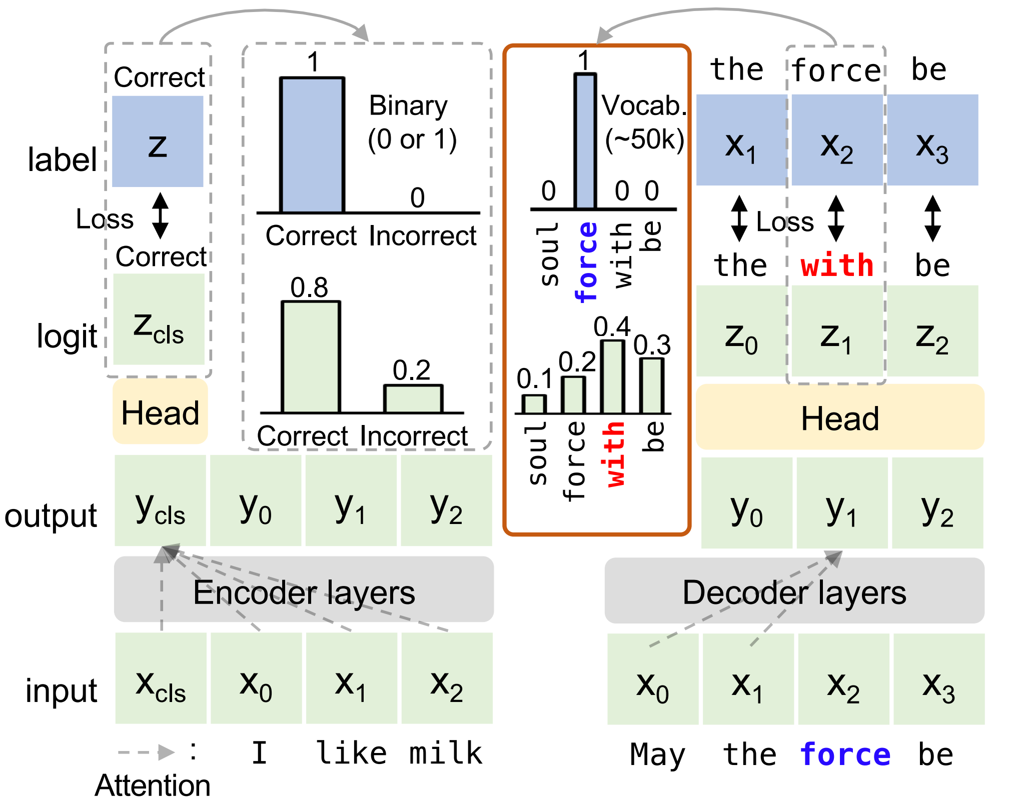
L2L QAT-KD PPL (↓)
W2A16: 20.47 (+2.30)

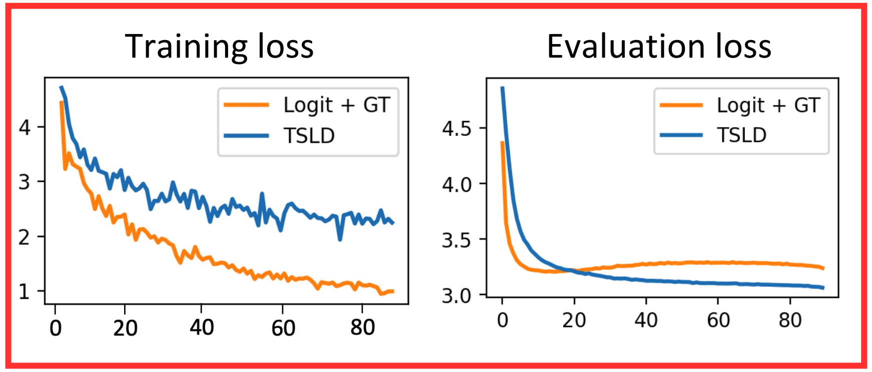
Logit QAT-KD PPL (↓)
W2A16: **18.86 (+0.61)**

Logit KD: memory-efficient and natural choice for GLM QAT

Challenge 2. How to Exploit Ground-Truth for Language Modeling in QAT?

- "Employing GT Loss in QAT-KD adversely impacts the performance of decoder!" [1]
- **Challenge) GT Loss is employed with KD in Decoder QAT, <u>overfitting is observed!</u>**

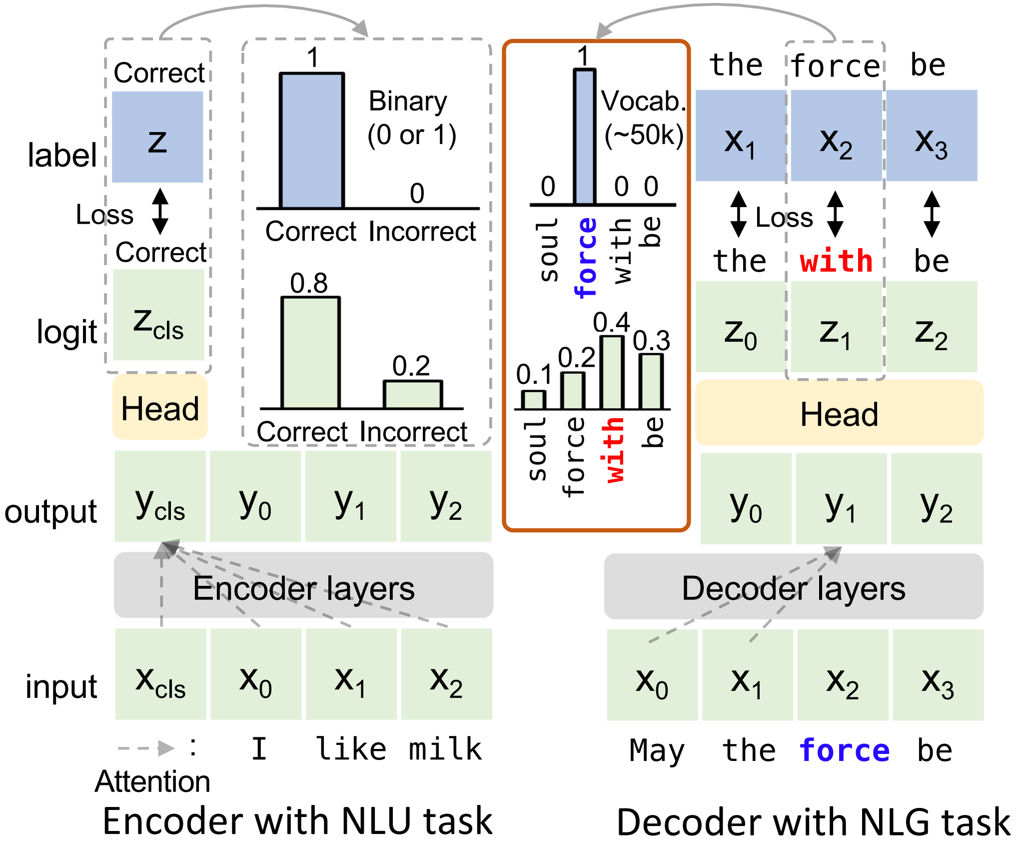


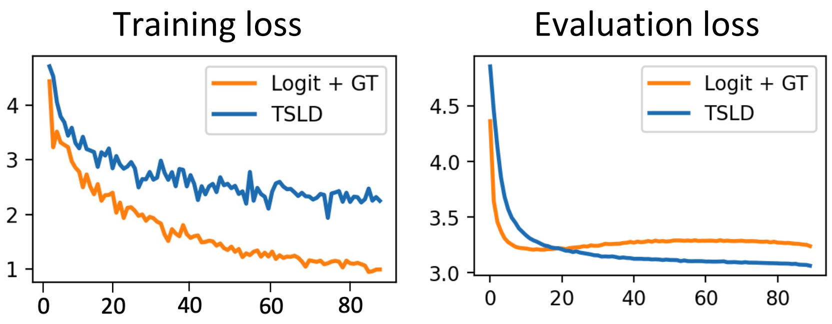


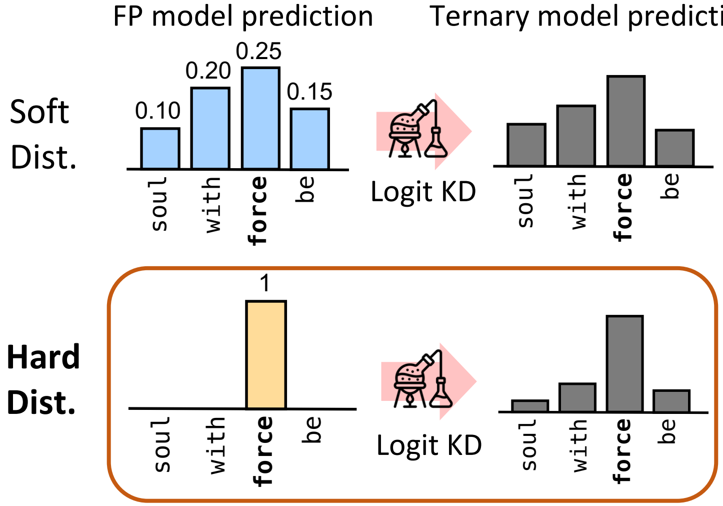
[1] Tao et al, *"Compression of Generative Pre-trained Language Models via Quantization"*, ACL 2022

Challenge 2. How to Exploit Ground-Truth for Language Modeling in QAT?

- "Employing GT Loss in QAT-KD adversely impacts the performance of decoder!" [1]
- **Challenge) GT Loss is employed with KD in Decoder QAT, overfitting is observed!**







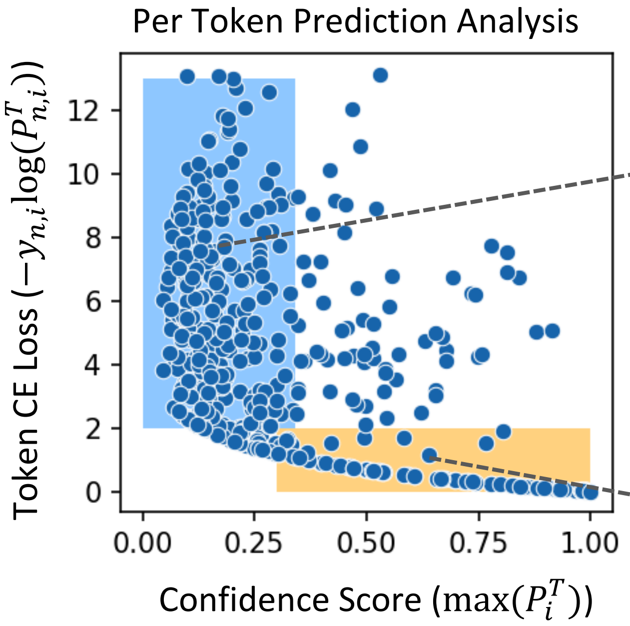
Potential overlap between Logit KD and GT Loss...

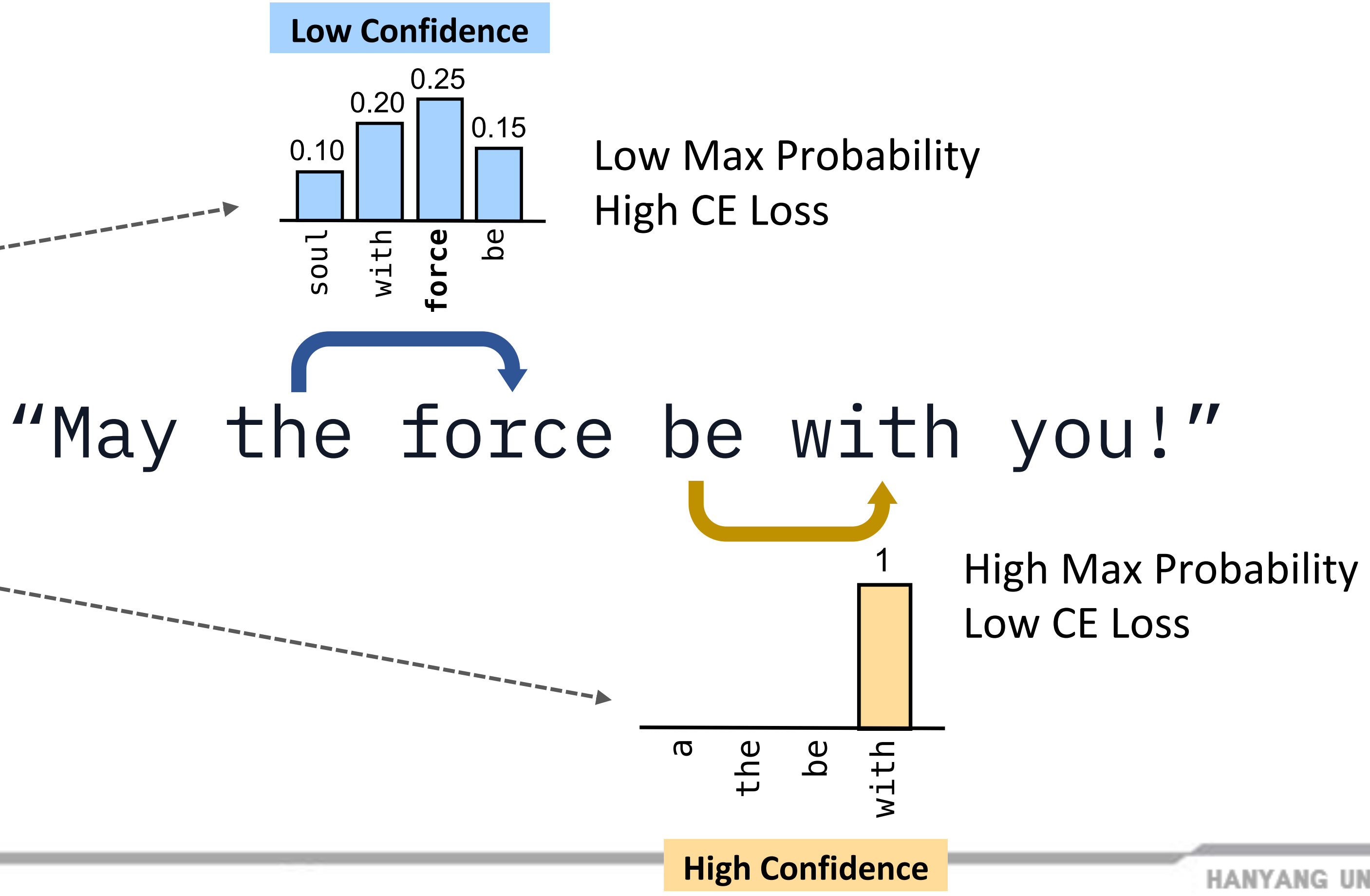
AI Hardware and Algorithm Lab

[1] Tao et al, "Compression of Generative Pre-trained Language Models via Quantization", ACL 2022

Token-Scaled Logit Distillation for Avoiding Overfitting with GT Loss

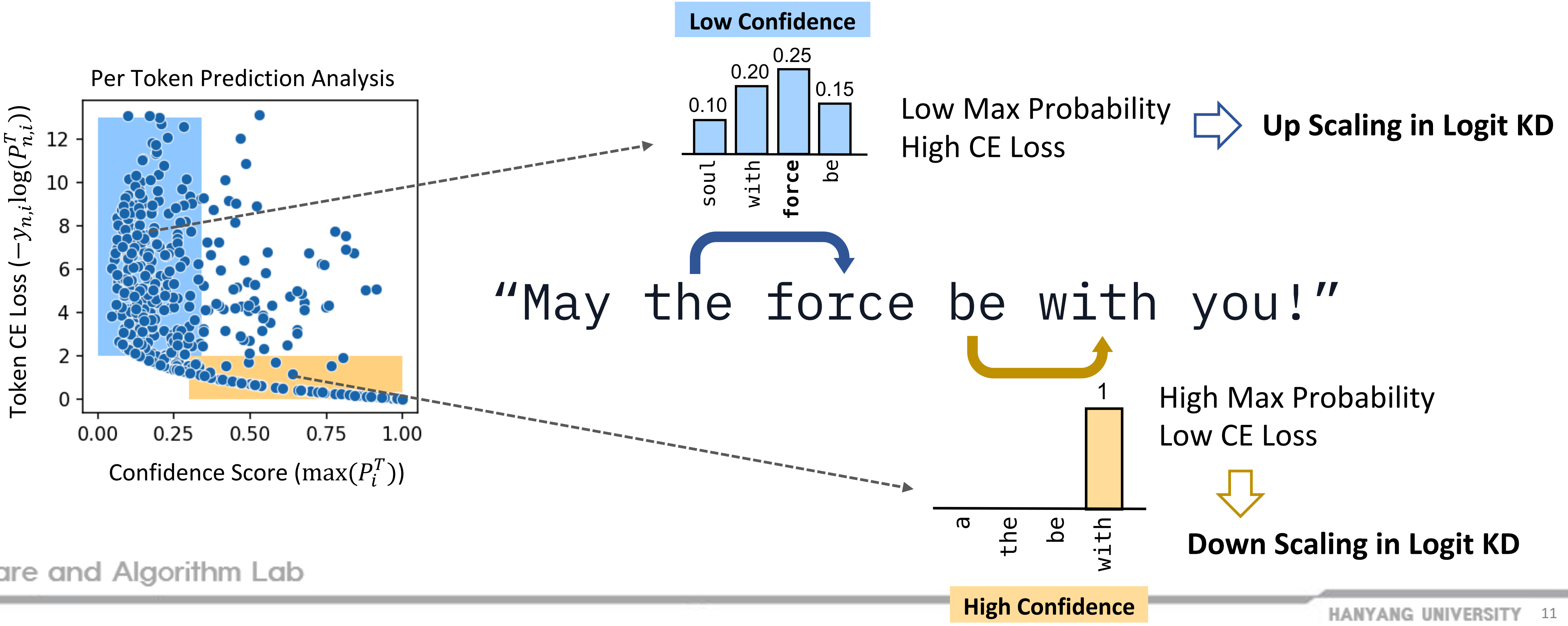
- Per-token prediction analysis -> **Token Confidence Demarcation: High Conf/Low Conf**
  - **High Conf**. : High max probability **with low CE loss (overlap with GT Loss)**
  - **Low Conf**. : Low max probability **with high CE loss (rich soft label information)**





Token-Scaled Logit Distillation for Avoiding Overfitting with GT Loss

- Per-token prediction analysis -> **Token Confidence Demarcation: High Conf/Low Conf**
  - **High Conf.** : High max probability with low CE loss (overlap with GT Loss) -> **Down Scaling**
  - **Low Conf. :** Low max probability with high CE loss (rich soft label information) -> **Up Scaling**

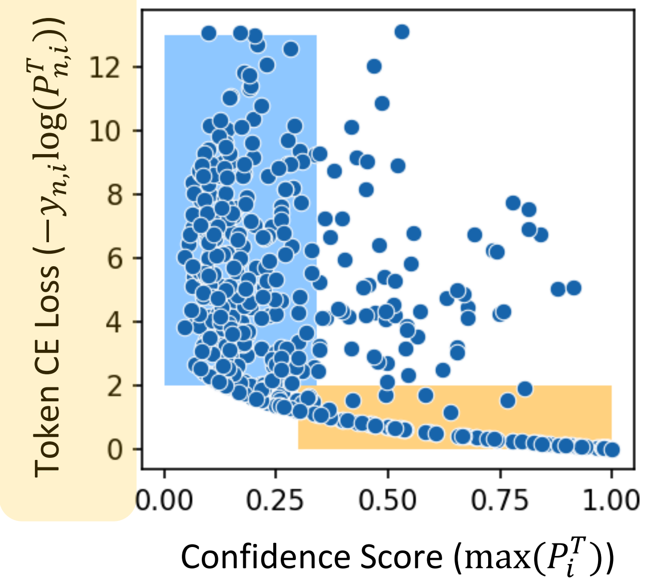


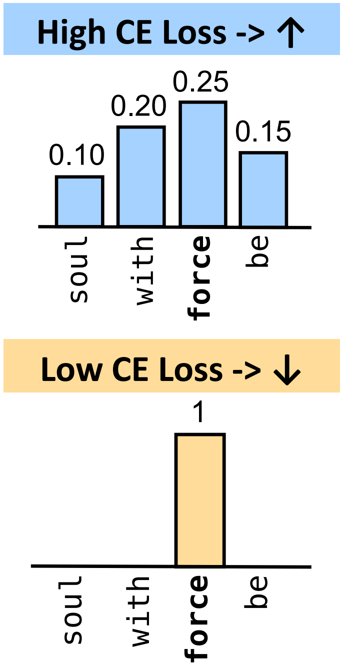
Token-Scaled Logit Distillation for Avoiding Overfitting with GT Loss

- **Token-Scaled Logit Distillation (TSLD)**
  - Apply dynamic reweighting to Logit KD: (↓) reduce overfitting + (↑) superior learning from teacher

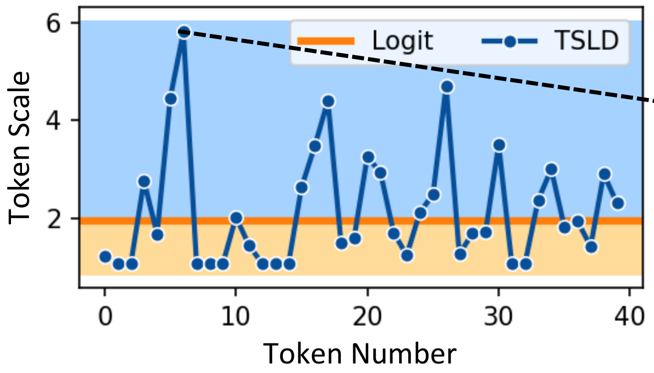
$$L_{TSLD} = \sum_{n=1}^{N} \left(\underbrace{\textbf{softmax}\left(-\sum_{i=1}^{V} y_{n,i}\, log(P^{T}_{n,i})\right)}_{\textbf{Token-Scaling}} \circ \underbrace{\sum_{i=1}^{V} -P^{T}_{n,i}\log(P^{S}_{n,i})}_{\text{Logit KD}} \right)$$

Per Token Prediction Analysis

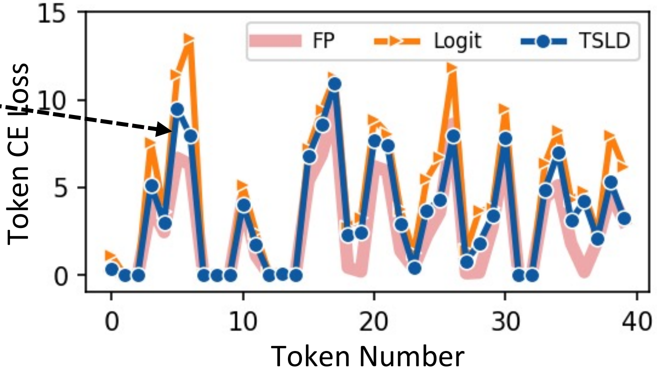




Per Token Scale (Logit KD vs TSLD)



Per Token CE Loss (Logit KD vs TSLD)



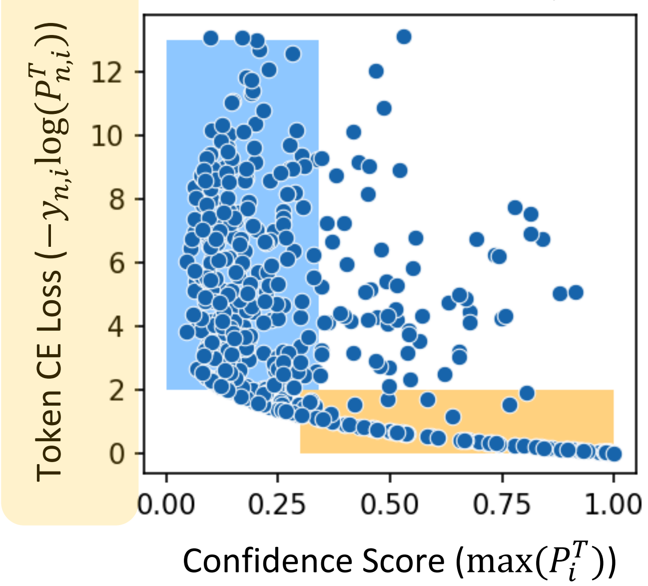
Up-Scaling => Further lowering token-wise CE Loss!

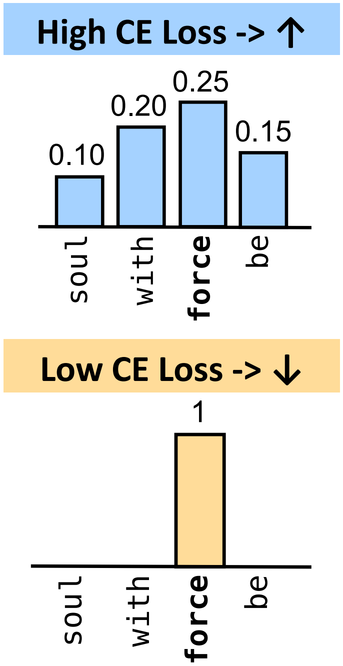
Token-Scaled Logit Distillation for Avoiding Overfitting with GT Loss

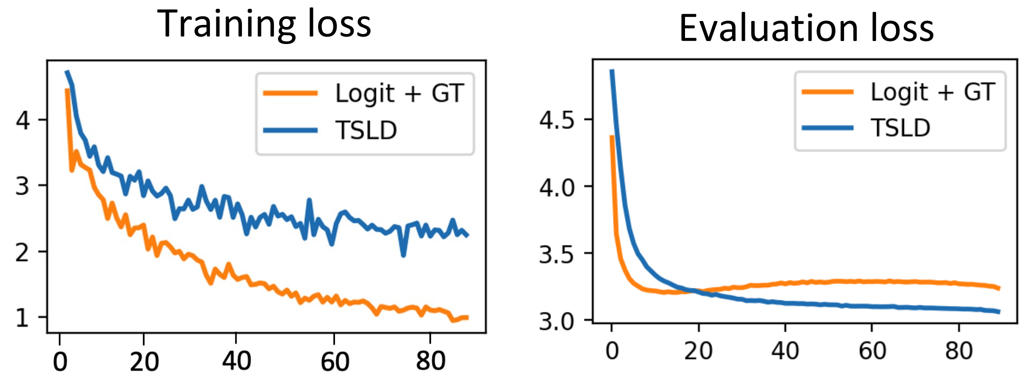
- **Token-Scaled Logit Distillation (TSLD)**
  - Apply dynamic reweighting to Logit KD: (↓) reduce overfitting + (↑) superior learning from teacher

$$L_{TSLD} = \sum_{n=1}^{N}\left(\underbrace{\mathbf{softmax}\left(-\sum_{i=1}^{V} y_{n,i}\,log\left(P^{T}_{n,i}\right)\right)}_{\textbf{Token-Scaling}} \circ \underbrace{\sum_{i=1}^{V} -P^{T}_{n,i}\log\left(P^{S}_{n,i}\right)}_{\text{Logit KD}}\right)$$

Per Token Prediction Analysis







Down-Scaling => Avoid overfitting!

Effectiveness of TSLD (1): Language Modeling

- Evaluation on Language Modeling Task over 0.1B to 6.7B GLMs
  - 4-bit: QAT methods outperforms PTQ (OPTQ) methods, TSLD offers the lowest perplexity
  - 2-bit (ternary): L2L KD shows significant perplexity degradation and Logit+GT suffers from overfitting.
  - **TSLD outperforms every KD methods across all model sizes.**

Perplexity evaluation in language modeling with 0.1B to 6.7B GLMs

Precision	Quantization Method	Optimization Method	GPT-2 0.1B	GPT-2 0.3B	GPT-2 0.8B	GPT-2 1.5B	OPT 0.1B	OPT 1.3B	OPT 2.7B	OPT 6.7B
FP16 baseline			20.91	18.21	15.20	14.26	18.17	13.75	11.43	10.21
W4A16	PTQ	OPTQ [13]	22.41	19.35	17.26	15.86	19.75	14.30	11.82	11.73
W4A16	QAT	Logit [32]	20.98	18.54	16.79	15.42	17.60	**13.73**	11.82	11.20
W4A16	QAT	Logit+GT	21.51	18.58	15.49	14.89	19.63	15.03	12.58	11.78
W4A16	QAT	TSLD	**19.95**	**17.53**	**15.32**	**14.50**	**17.45**	13.90	**11.59**	**11.00**
W2A16	QAT	L2L+Logit [44]	23.79	21.21	17.80	15.82	20.47	17.62	14.67	11.75
W2A16	QAT	Logit [32]	22.84	19.87	16.46	15.27	18.86	14.80	12.26	11.33
W2A16	QAT	Logit+GT	23.80	20.20	17.77	16.52	21.62	16.41	13.20	12.41
W2A16	QAT	TSLD	**21.74**	**18.57**	**16.14**	**15.02**	**18.58**	**14.60**	**11.97**	**11.17**

Table 1: Perplexity comparison in GPT-2 and OPT series across various model sizes (0.1B to 6.7B) on the PTB dataset with QAT-KD (tensor-wise) and PTQ (channel-wise) quantization methods

Effectiveness of TSLD (2): Reasoning and NLU Task Accuracy

- Evaluation of reasoning task and natural language understanding tasks over OPT, GPT-Neo, and LLaMA
  - With "task" accuracy, including GT Loss in KD outperforms Logit KD only.
  - **TSLD achieves better task accuracy thanks to avoiding overfitting from ground-truth**

QAT KD Method	PIQA		OpenbookQA		ARC_easy		ARC_challenge		GSM8K	
	ACC (↑)	PPL (↓)	ACC (↑)	PPL (↓)	ACC (↑)	PPL (↓)	ACC (↑)	PPL (↓)	ACC (↑)	PPL (↓)
OPT-2.7B FP16	76.71	10.91	49.60	26.16	66.12	7.41	37.20	8.96	20.39	2.07
Logit [24]	74.32	11.69	45.40	29.41	58.92	9.05	31.91	12.38	20.02	**2.03**
GT+Logit	74.97	12.10	46.20	31.08	58.84	8.66	32.16	12.04	19.56	2.12
TSLD	**75.62**	**11.35**	**46.81**	**28.93**	**59.39**	**8.12**	**33.45**	**11.05**	**20.24**	**2.03**

Model	GPT-Neo-1.3B	OPT-6.7B		LLaMA-7B		
QAT KD	PTB (PPL)	GSM8K (ACC/PPL)		PTB (PPL)	GSM8K (ACC/PPL)	
FP16	17.62 (↓)	22.52 (↑)	1.89 (↓)	8.76 (↓)	30.25 (↑)	1.47 (↓)
Logit [24]	21.01	21.08	**1.93**	12.22	25.47	**1.52**
TSLD	**19.27**	**24.49**	2.14	**11.60**	**26.23**	**1.52**

Reasoning Task evaluation with OPT series, GPT-Neo-1.3B, and LLaMA-7B

Precision	QAT KD Method	CoLA		MRPC		SST-2		RTE	
		ACC (↑)	PPL (↓)	ACC (↑)	PPL (↓)	ACC (↑)	PPL (↓)	ACC (↑)	PPL (↓)
OPT-1.3B FP16		61.03	1.34	81.92	2.58	94.26	2.00	76.53	3.94
W4A16	OPTQ [14]	**54.61**	**1.36**	**80.14**	**2.43**	**95.07**	**2.02**	**56.32**	**3.96**
	AWQ [13]	13.63	1.45	66.42	3.49	94.26	**2.02**	54.51	4.72
	Logit [24]	50.76 ±2.35	1.36	81.94 ±1.48	2.62	93.57 ±0.23	2.09	75.23 ±0.83	4.34
	GT+Logit	54.07 ±0.34	**1.34**	83.17 ±0.51	2.60	93.34 ±0.22	2.11	75.31 ±1.07	4.09
	TSLD	**56.33** ±0.98	**1.34**	**83.33** ±1.22	**2.52**	**94.05** ±0.19	**2.04**	**75.97** ±0.31	**4.05**
W2A16	Logit [24]	48.72 ±2.68	1.37	81.62 ±0.62	2.79	93.08 ±0.35	2.11	74.15 ±1.36	4.72
	GT+Logit	50.10 ±1.38	**1.34**	82.10 ±0.99	2.65	92.77 ±0.28	2.14	73.79 ±1.16	4.44
	TSLD	**54.47** ±1.47	**1.34**	**82.20** ±0.94	**2.63**	**93.92** ±0.29	**2.06**	**75.31** ±0.54	**4.36**

NLU task evaluation with OPT-1.3B (language modeling fine-tuning employed)

Thank You!

For more question and discussion, please visit poster session 3 #536. 🤗

Paper **Code**