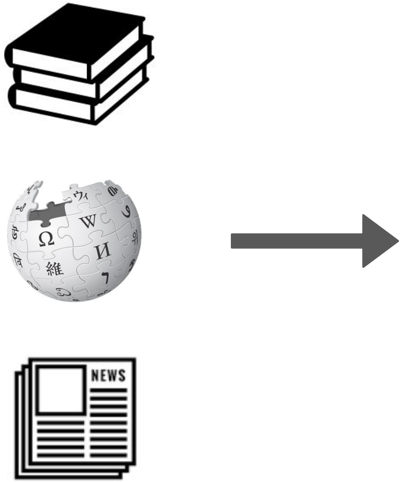
Efficient Transfer Learning with Large Language Models

Yoon Kim
MIT
(work with Demi Guo, Alexander Rush, Hunter Lang, Monica Agrawal, David Sontag)

Language Models

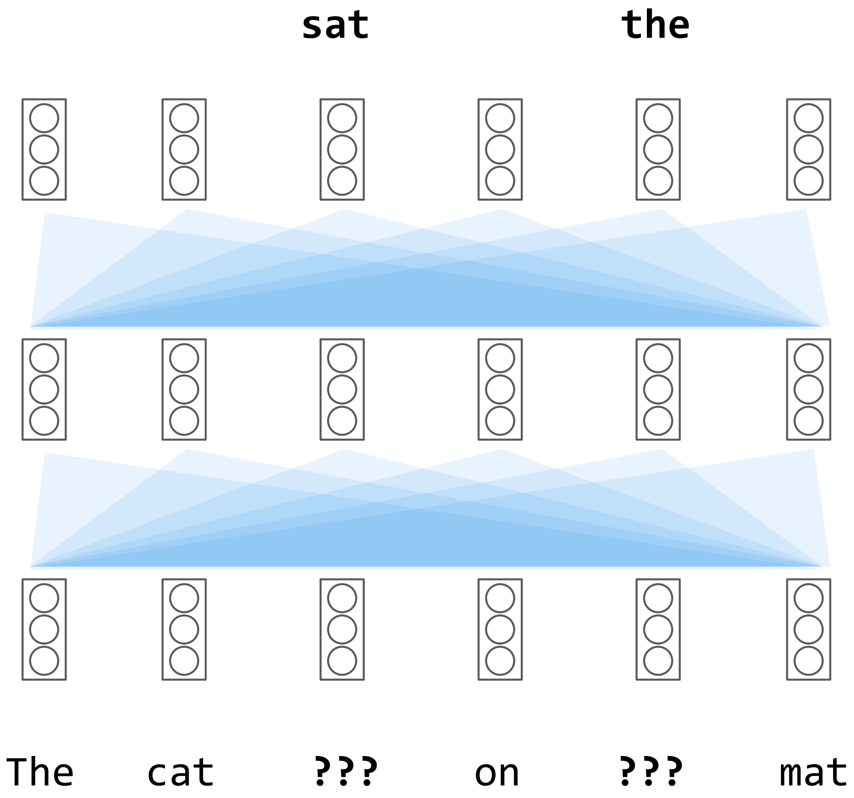


I see a beautiful city and a brilliant …

Albert Camus was a French philosopher, author …

GameStop stock rises after chairman buys …

Masked Language Models

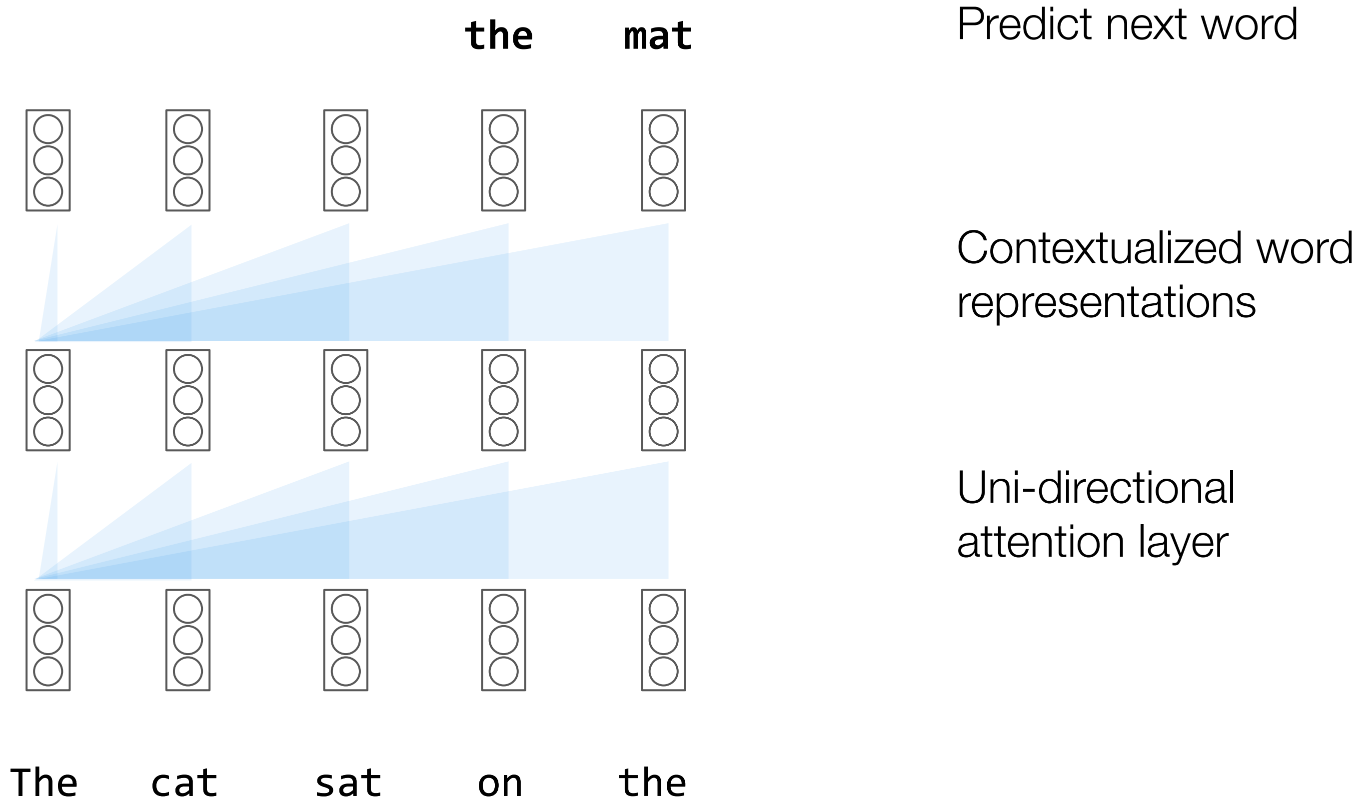


Predict masked word

Contextualized word representations

Bi-directional attention layer

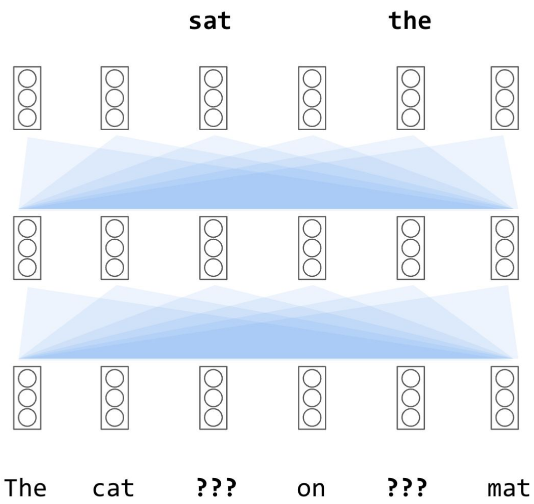
Autoregressive Language Models



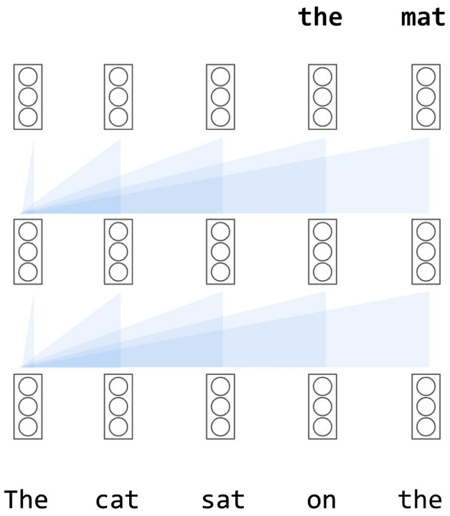
Language Modeling

$$\max_{\theta} \prod_{(w,c)\in\mathcal{D}} p_\theta(w \mid c)$$

$w =$ word
$c =$ context



$w =$ masked word
$c =$ surrounding words



$w =$ next word
$c =$ previous words

Language Modeling Objective

Language models can implicitly capture much linguistic/world knowledge through their parameters.

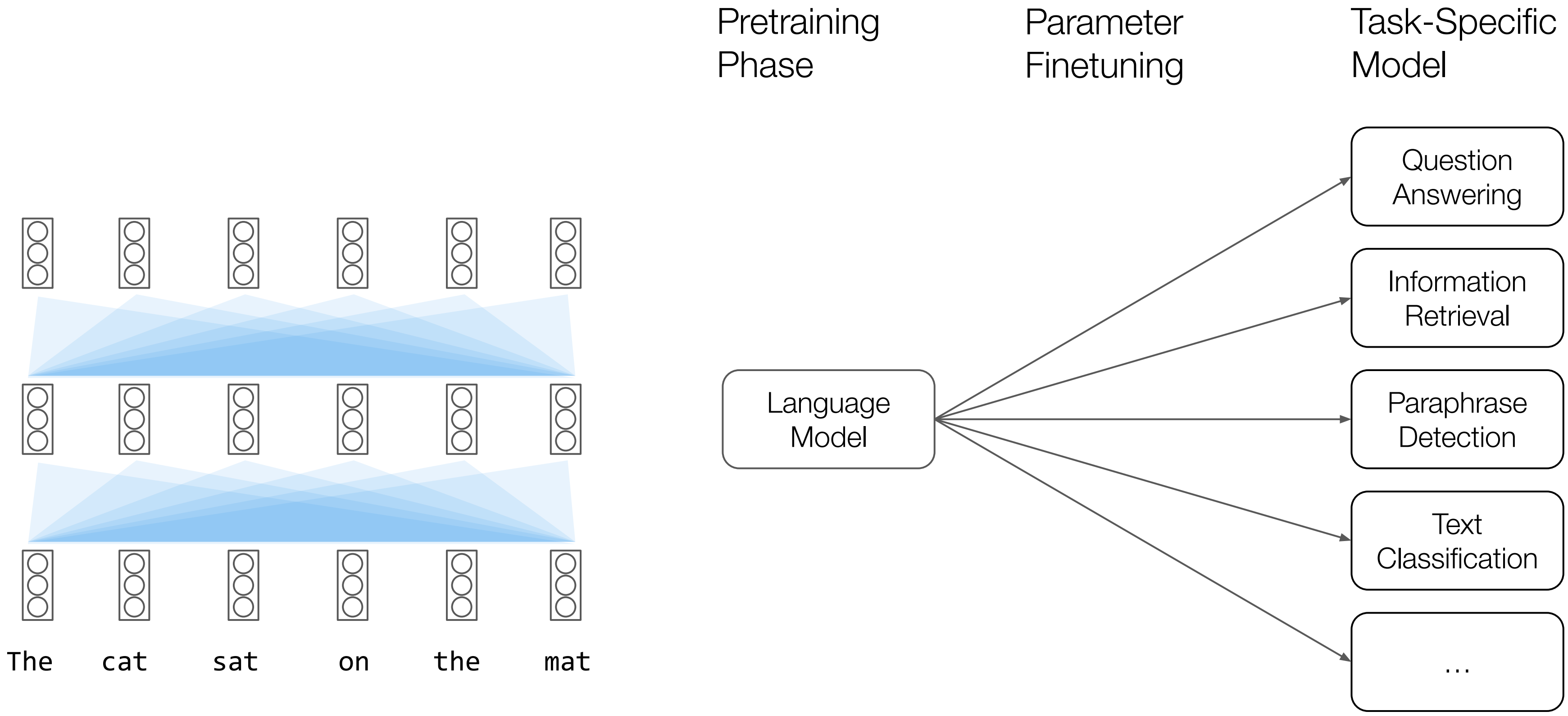
$$\max_{\theta} \prod_{(w,c)\in\mathcal{D}} p_{\theta}(w \mid c) \longrightarrow \theta$$

$w =$ word

$c =$ context

Transfer learning paradigm: finetuning / prompting.

Transfer Learning via Finetuning

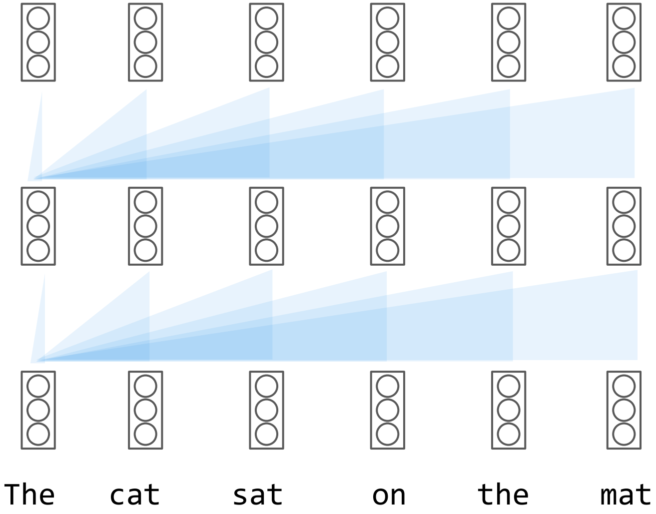


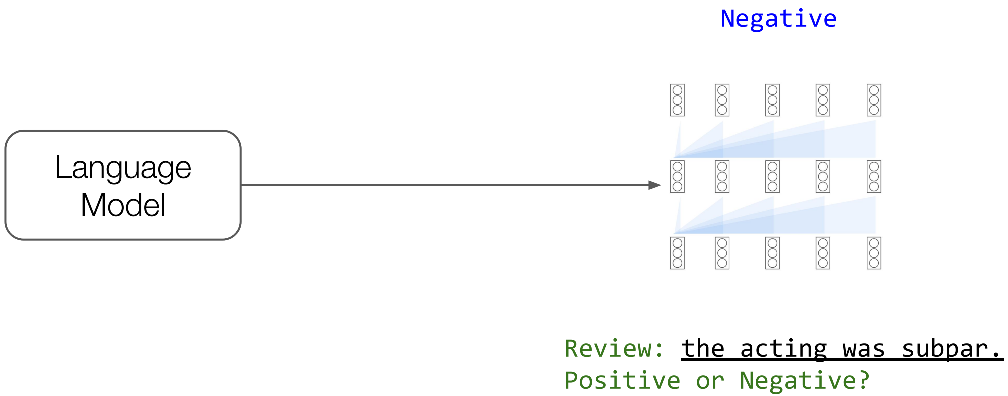
Transfer Learning via Prompting

Pretraining Phase

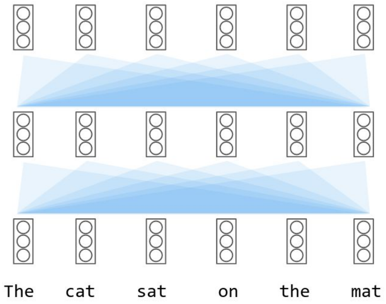
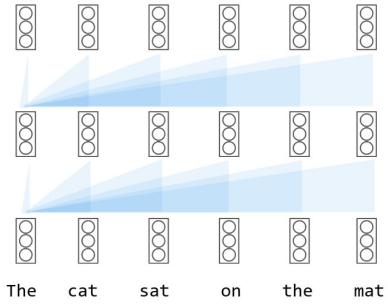
Conditioning via Language "Prompts"

Task-Specific Model

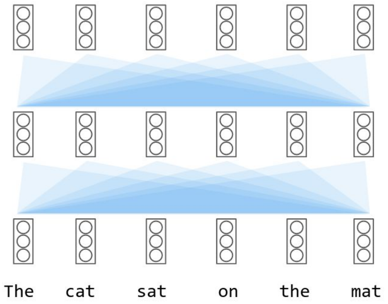




Transfer Learning with Language Models

	Examples	Size	Adaptation	Labeled data	Inference
	BERT RoBERTa XLNet BART T5	100M-10B	Fine-tuning	>16	Fast
	GPT-3 GLaM T0 FLAN PaLM	10B-500B	Prompting	<16	Slow

Transfer Learning with Language Models



Examples	Size	Adaptation	Labeled data	Inference
BERT RoBERTa XLNet BART T5	100M-10B	Fine-tuning	>16	Fast

✓ Good performance and reasonably fast inference.

✗ Task-specific parameters ⇒ memory does not scale well to multiple tasks.

✗ Still requires nontrivial amounts of labeled data.

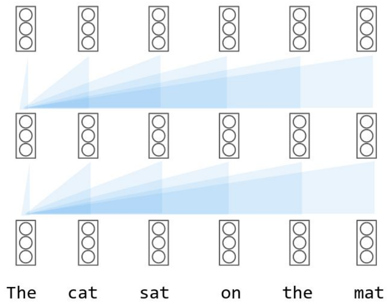
Transfer Learning with Language Models

	Examples	Size	Adaptation	Labeled data	Inference

✓ Pretrained parameters remain fixed.

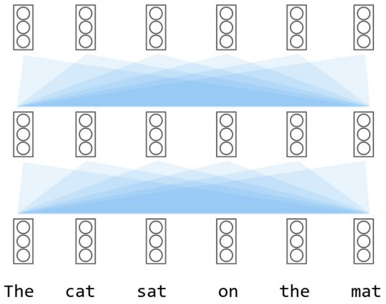
✓ Good few-shot and zero-shot performance.

✗ Prompting capabilities only emerge when model sizes are large enough ⇒ inference is slow.



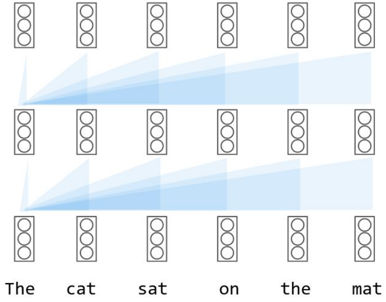
	Examples	Size	Adaptation	Labeled data	Inference
	GPT-3 GLaM T0 FLAN PaLM	10B-500B	Prompting	<16	Slow

Efficient Transfer Learning with Language Models



Memory Efficiency:
“Parameter-Efficient Transfer Learning with Diff Pruning”


(with Demi Guo, Alexander Rush; ACL ’21)

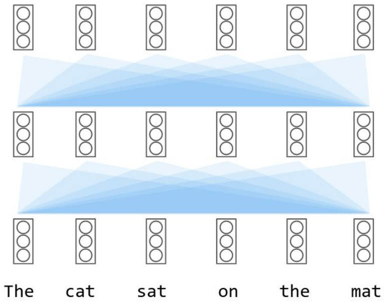


Inference Efficiency:
“Co-training Improves Prompt-based Learning for Large Language Models”



(with Hunter Lang, Monica Agrawal, David Sontag; ICML ’22)

Efficient Transfer Learning with Language Models



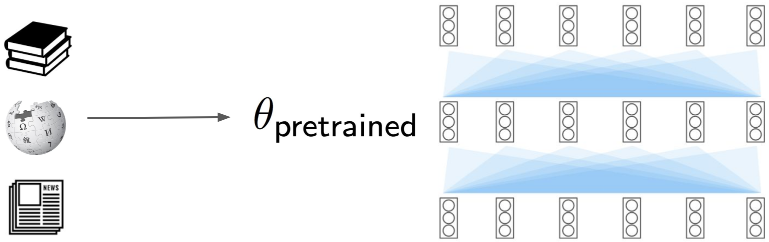
Memory Efficiency:
"Parameter-Efficient Transfer Learning with Diff Pruning"


(with Demi Guo, Alexander Rush; ACL '21)

Inference Efficiency:
"Co-training Improves Prompt-based Learning for Large Language Models"

(with Hunter Lang, Monica Agrawal, David Sontag; ICML '22)

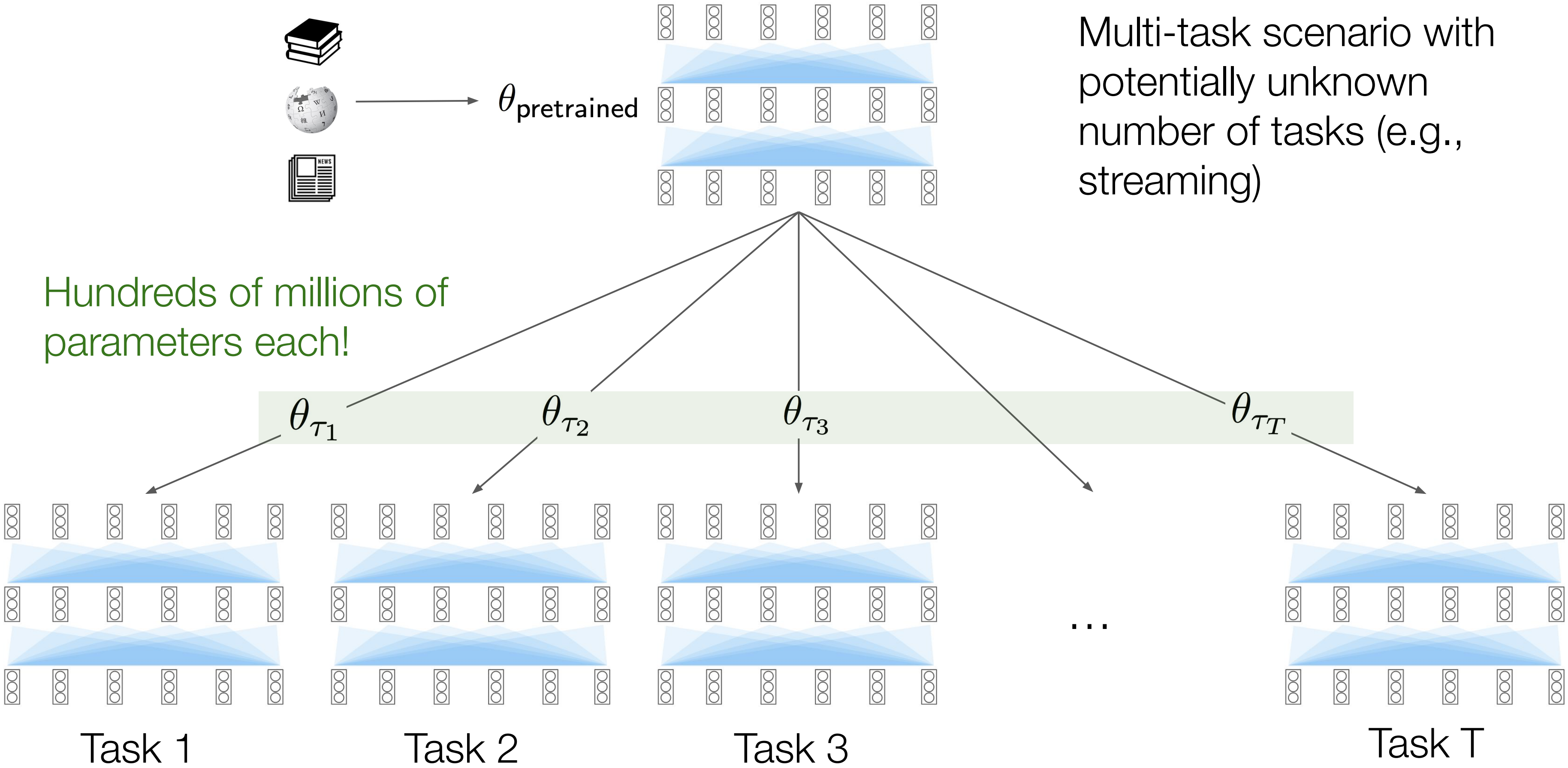
Transfer Learning via Full Fine-tuning



Transfer Learning via Full Fine-tuning

Multi-task scenario with potentially unknown number of tasks (e.g., streaming)

Hundreds of millions of parameters each!

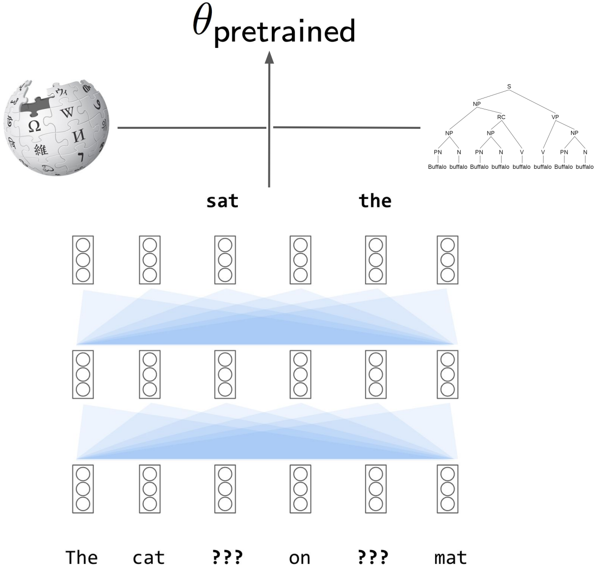


Transfer Learning via Full Fine-tuning

- Full fine-tuning: need to store full set of parameters for each task ⇒ hard to scale to multiple tasks.

- Model already learns linguistic and world knowledge through pretraining ⇒ unnecessary/wasteful to fine-tune all parameters.

  (Parameter-inefficiency)



Existing Approaches for Parameter Efficiency

- Model compression:
  - Pruning [Goden et al. ’20, Sajjad et al. ’20, Chen et al. ’20]
  - Distillation [Sanh et al. ’19, Sun et al. ’20, Jiao et al. ’20]

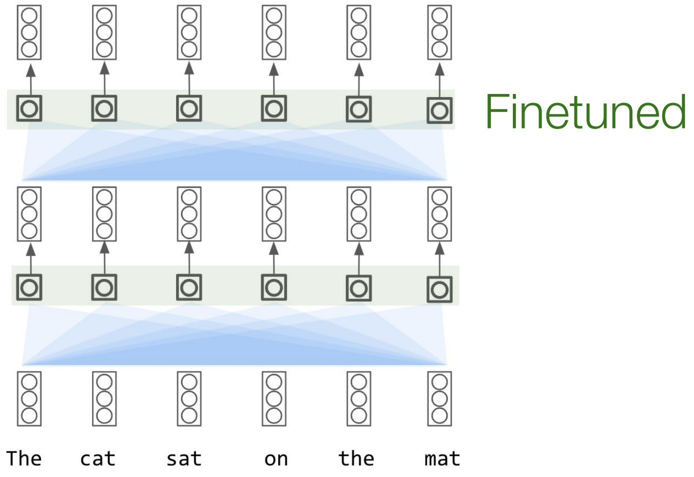
  Still requires 10%-30% of the full parameters to maintain performance.

Existing Approaches for Parameter Efficiency

- Model compression:
  - Pruning [Goden et al. ’20, Sajjad et al. ’20, Chen et al. ’20]
  - Distillation [Sanh et al. ’19, Sun et al. ’20, Jiao et al. ’20]

  Still requires 10%-30% of the full parameters to maintain performance.

- Adapters [Houlsby et al. ’19]:
  - Small narrow layers that are inserted in between wider model layers.
  - Pretrained model remains fixed, only the adapters are fine-tuned for each task. (One adapter per task).
  - Only requires 2%-4% new parameters per task!



Diff Pruning

- Learn an *extension* to the existing pretrained model (which remains fixed).

- Model extension is parameterized as a vector ("diff<u>erence</u> vector") that additively modifies pretrained parameters.

$$
\begin{aligned}
\theta_{\tau_1} &= \theta_{\text{pretrained}} + \delta_{\tau_1} \\
\theta_{\tau_2} &= \theta_{\text{pretrained}} + \delta_{\tau_2} \\
\theta_{\tau_3} &= \theta_{\text{pretrained}} + \delta_{\tau_3} \\
&\vdots \\
\theta_{\tau_T} &= \theta_{\text{pretrained}} + \delta_{\tau_T}
\end{aligned}
$$

Diff Pruning

- Learn an *extension* to the existing pretrained model (which remains fixed).

- Model extension is parameterized as a vector ("difference vector") that additively modifies pretrained parameters.

$$
\begin{aligned}
\theta_{\tau_1} &= \theta_{\text{pretrained}} + \delta_{\tau_1} \\
\theta_{\tau_2} &= \theta_{\text{pretrained}} + \delta_{\tau_2} \\
\theta_{\tau_3} &= \theta_{\text{pretrained}} + \delta_{\tau_3} \\
&\vdots \\
\theta_{\tau_T} &= \theta_{\text{pretrained}} + \delta_{\tau_T}
\end{aligned}
$$

If the extension (diff vector) is sparse, then additional memory per task will be marginal.

Diff Pruning Objective

- For each task τ:

$$\min_{\delta_\tau} \sum_{n=1}^{N} -\log p(y^{(n)} \mid x^{(n)}; \theta_{\text{pretrained}} + \delta_\tau) + \lambda R(\delta_\tau)$$

Diff Pruning Objective

- For each task $\mathcal{T}$:

$$\min_{\delta_\mathcal{T}} \sum_{n=1}^{N} -\log p(y^{(n)} \mid x^{(n)}; \theta_{\text{pretrained}} + \delta_\mathcal{T}) + \lambda R(\delta_\mathcal{T})$$

Task-specific negative log likelihood

Regularizer on diff vector

- If regularizer can learn a sparse diff vector such that $\|\delta_\mathcal{T}\|_0 \ll \|\theta_{\text{pretrained}}\|_0$ then we only need a few additional parameters per task!

Differentiable Sparse Regularizer [Louizos et al. ’18]

Original Objective

$$\min_{\delta_\tau} \; L(\mathcal{D}_\tau, \theta_{\textsf{pretrained}} + \delta_\tau) + \lambda R(\delta_\tau)$$

L_0-norm regularizer

$$R(\delta_\tau) = \sum_{i=1}^{d} \mathbb{1}\{\delta_{\tau,i} \neq 0\}$$

Not amenable to gradient-based optimization

Differentiable Sparse Regularizer [Louizos et al. '18]

Original Objective $$\min_{\delta_\tau} L(\mathcal{D}_\tau, \theta_{\textsf{pretrained}} + \delta_\tau) + \lambda R(\delta_\tau)$$

L_0-norm regularizer $$R(\delta_\tau) = \sum_{i=1}^{d} \mathbb{1}\{\delta_{\tau,i} \neq 0\}$$

(Still) not amenable to gradient-based optimization

Decompose diff vector $$\delta_\tau = z_\tau \odot w_\tau, \qquad z_\tau \in \{0,1\}^d,\ w_\tau \in \mathbb{R}^d$$

Reparameterized Objective $$\min_{z_\tau, w_\tau} L(\mathcal{D}_\tau, \theta_{\textsf{pretrained}} + z_\tau \odot w_\tau) + \lambda R(z_\tau \odot w_\tau)$$

Differentiable Sparse Regularizer [Louizos et al. '18]

Original Objective $\min_{\delta_\tau} L(\mathcal{D}_\tau, \theta_{\textsf{pretrained}} + \delta_\tau) + \lambda R(\delta_\tau)$

L0-norm regularizer $R(\delta_\tau) = \sum_{i=1}^{d} \mathbb{1}\{\delta_{\tau,i} \neq 0\}$

Decompose diff vector $\delta_\tau = z_\tau \odot w_\tau, \qquad z_\tau \in \{0,1\}^d,\ w_\tau \in \mathbb{R}^d$

Lower bound $\min_{\alpha_\tau, w_\tau} \mathbb{E}_{z_\tau \sim p(z_\tau;\, \alpha_\tau)} \left[L(\mathcal{D}_\tau, \theta_{\textsf{pretrained}} + z_\tau \odot w_\tau) + \lambda R(z_\tau \odot w_\tau)\right]$

Optimize over distribution parameterized by α_τ:

$$p(z_\tau;\, \alpha_\tau) = \prod_{i=1}^{d} \sigma(\alpha_{\tau,i})^{z_{\tau,i}} \times (1 - \sigma(\alpha_{\tau,i}))^{1 - z_{\tau,i}}$$

Issue: Tractable optimization requires policy gradients.

Differentiable Sparse Regularizer [Louizos et al. ’18]

Original Objective $\min_{\delta_\tau} L(\mathcal{D}_\tau, \theta_{\text{pretrained}} + \delta_\tau) + \lambda R(\delta_\tau)$

L_0-norm regularizer $R(\delta_\tau) = \sum_{i=1}^{d} \mathbb{1}\{\delta_{\tau,i} \neq 0\}$

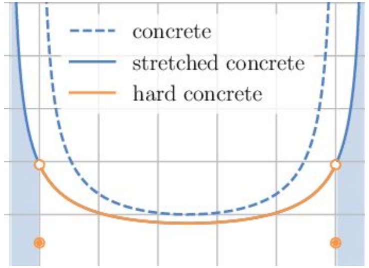
Decompose diff vector $\delta_\tau = z_\tau \odot w_\tau, \quad z_\tau \in \{0,1\}^d, \; w_\tau \in \mathbb{R}^d$

Lower bound $\min_{\alpha_\tau, w_\tau} \mathbb{E}_{z_\tau \sim p(z_\tau; \alpha_\tau)} [L(\mathcal{D}_\tau, \theta_{\text{pretrained}} + z_\tau \odot w_\tau) + \lambda R(z_\tau \odot w_\tau)]$

Continuous relaxation $z_\tau \in \{0,1\}^d \rightarrow \tilde{z}_\tau \in [0,1]^d$

$$
\begin{aligned}
u &\sim U[0,1] \\
s_\tau &= \sigma(\log u - \log(1-u) + \alpha_\tau) \\
\bar{s}_\tau &= (r - l) \times s_\tau + l \\
\tilde{z}_\tau &= \min(1, \max(0, \bar{s}_\tau))
\end{aligned}
$$

Stretched Hard-Concrete distribution [Louizos et al. ’18]



Differentiable Sparse Regularizer [Louizos et al. '18]

Original Objective $$\min_{\delta_\tau} \; L(\mathcal{D}_\tau, \theta_{\textsf{pretrained}} + \delta_\tau) + \lambda R(\delta_\tau)$$

L_0-norm regularizer $$R(\delta_\tau) = \sum_{i=1}^{d} \mathbb{1}\{\delta_{\tau,i} \neq 0\}$$

Decompose diff vector $$\delta_\tau = z_\tau \odot w_\tau, \qquad z_\tau \in \{0,1\}^d, \; w_\tau \in \mathbb{R}^d$$

Lower bound $$\min_{\alpha_\tau, w_\tau} \; \mathbb{E}_{z_\tau \sim p(z_\tau;\, \alpha_\tau)} \left[L(\mathcal{D}_\tau, \theta_{\textsf{pretrained}} + z_\tau \odot w_\tau) + \lambda R(z_\tau \odot w_\tau) \right]$$

Continuous relaxation $$z_\tau \in \{0,1\}^d \rightarrow \tilde{z}_\tau \in [0,1]^d$$

Reparameterization trick
⇒ lower-variance gradient estimator.

$$\min_{\alpha_\tau, w_\tau} \; \mathbb{E}_{u \sim U[0,1]} \left[L(\mathcal{D}_\tau, \theta_{\textsf{pretrained}} + \tilde{z}_\tau \odot w_\tau) + \lambda R(\tilde{z}_\tau \odot w_\tau) \right]$$

Differentiable Sparse Regularizer [Louizos et al. '18]

Original Objective $$\min_{\delta_\tau} L(\mathcal{D}_\tau, \theta_{\textsf{pretrained}} + \delta_\tau) + \lambda R(\delta_\tau)$$

L_0-norm regularizer $$R(\delta_\tau) = \sum_{i=1}^{d} \mathbb{1}\{\delta_{\tau,i} \neq 0\}$$

Decompose diff vector $$\delta_\tau = z_\tau \odot w_\tau, \qquad z_\tau \in \{0,1\}^d,\ w_\tau \in \mathbb{R}^d$$

Lower bound $$\min_{\alpha_\tau, w_\tau} \mathbb{E}_{z_\tau \sim p(z_\tau;\alpha_\tau)} \left[L(\mathcal{D}_\tau, \theta_{\textsf{pretrained}} + z_\tau \odot w_\tau) + \lambda R(z_\tau \odot w_\tau)\right]$$

Continuous relaxation $$z_\tau \in \{0,1\}^d \rightarrow \tilde{z}_\tau \in [0,1]^d$$

Reparameterization trick $$\min_{\alpha_\tau, w_\tau} \mathbb{E}_{u \sim U[0,1]} \left[L(\mathcal{D}_\tau, \theta_{\textsf{pretrained}} + \tilde{z}_\tau \odot w_\tau) + \lambda R(\tilde{z}_\tau \odot w_\tau)\right]$$

Closed-form solution for regularizer! $$\mathbb{E}_{u \sim U[0,1]}\left[R(\tilde{z}_\tau \odot w_\tau)\right] = \sum_{i=1}^{d} \sigma\left(\alpha_{\tau,i} - \log \frac{-l}{r}\right)$$

Diff Pruning

$$\min_{\alpha_\tau, w_\tau} \; \mathbb{E}_{u \sim U[0,1]} \left[L(\mathcal{D}_\tau, \theta_{\textsf{pretrained}} + \tilde{z}_\tau \odot w_\tau) \right] + \lambda \sum_{i=1}^{d} \sigma \left(\alpha_{\tau,i} - \log \frac{-l}{r} \right)$$

- After training α_τ should be very negative for many dimensions.
- Use this to get a sparse binary vector from:

$$p(z_\tau;\, \alpha_\tau) = \prod_{i=1}^{d} \sigma(\alpha_{\tau,i})^{z_{\tau,i}} \times (1 - \sigma(\alpha_{\tau,i}))^{1 - z_{\tau,i}}$$

- Final diff vector given by:

$$\delta_\tau = z_\tau \odot w_\tau, \qquad z_\tau \in \{0,1\}^d,\; w_\tau \in \mathbb{R}^d$$

Diff Pruning with Targeted Sparsity

$$\min_{\alpha_\tau, w_\tau} \mathbb{E}_{u \sim U[0,1]} \left[L(\mathcal{D}_\tau, \theta_{\text{pretrained}} + \tilde{z}_\tau \odot w_\tau) \right] + \lambda \sum_{i=1}^{d} \sigma \left(\alpha_{\tau,i} - \log \frac{-l}{r} \right)$$

$$\delta_\tau = z_\tau \odot w_\tau, \qquad z_\tau \in \{0,1\}^d, \;\; w_\tau \in \mathbb{R}^d$$

- Sparsity can be softly controlled by λ , but we often want *exact* sparsity control (e.g., memory budget).

- Targeted sparsity via projection onto L_0-ball (magnitude pruning):
  - Take the top *t*% of non-zero values of δ_τ based on magnitude.
  - Continue fine-tuning for a few epochs.

- Standard magnitude pruning on the *diff vector*.

Structured Diff Pruning

$$\min_{\alpha_\tau, w_\tau} \mathbb{E}_{u \sim U[0,1]} \left[L(\mathcal{D}_\tau, \theta_{\textsf{pretrained}} + \tilde{z}_\tau \odot w_\tau) \right] + \lambda \sum_{i=1}^{d} \sigma \left(\alpha_{\tau,i} - \log \frac{-l}{r} \right)$$

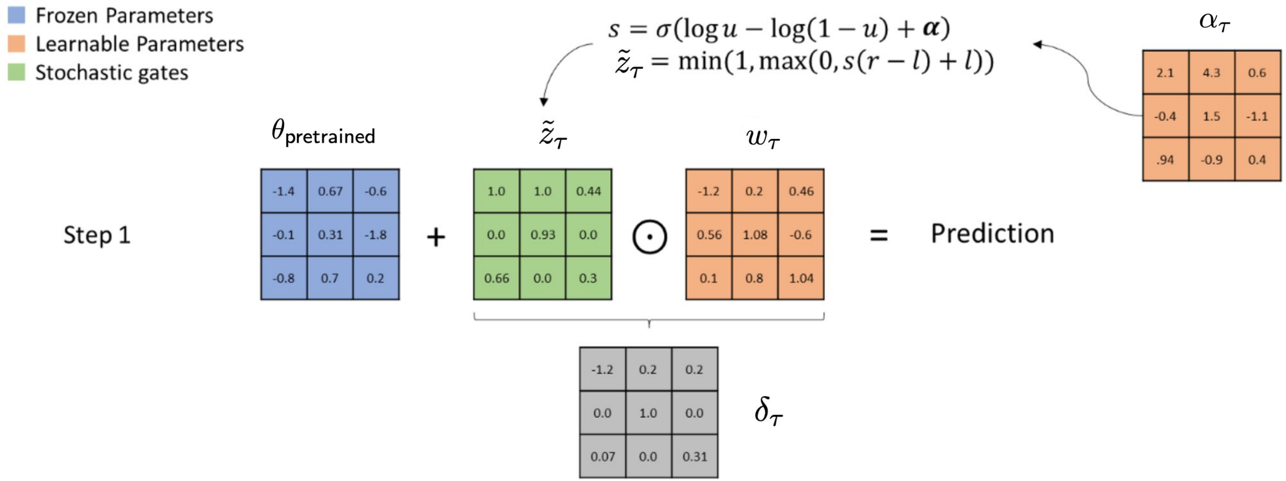
$$\delta_\tau = z_\tau \odot w_\tau, \qquad z_\tau \in \{0,1\}^d, \;\; w_\tau \in \mathbb{R}^d$$

- Partition each dimension into *groups* based on matrices/biases (393 groups for BERTLARGE):

$$\delta^j_{\tau,i} = z_{\tau,i} \times z^j_\tau \times w_{\tau,i}$$

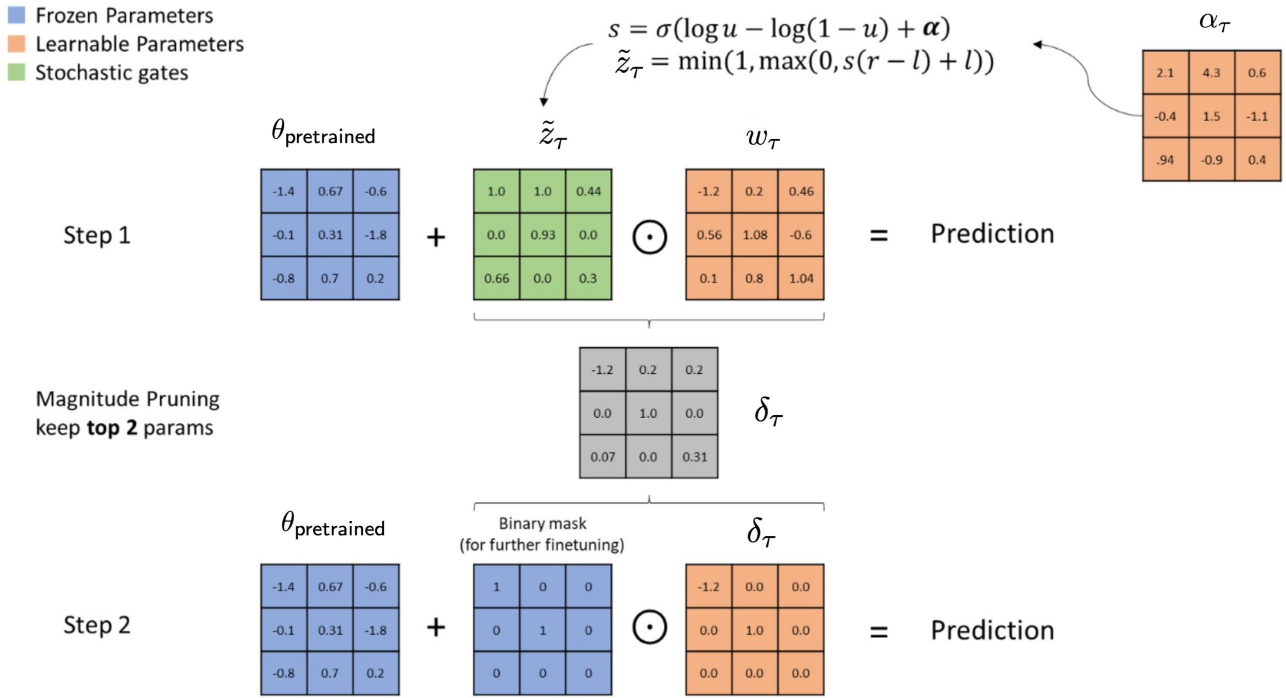
- Encourages entire groups to have zero diff vector.

Diff Pruning



(Image from https://medium.com/@lukas.hauzenberger/an-practical-introduction-to-diff-pruning-for-bert-4278ee4be750)

Diff Pruning



(Image from https://medium.com/@lukas.hauzenberger/an-practical-introduction-to-diff-pruning-for-bert-4278ee4be750)

Experiments

- Experiments on standard GLUE benchmark with $\text{BERT}_{\text{LARGE}}$.
- (Mostly) the same hyperparameters for all datasets.

Experiments

- Experiments on standard GLUE benchmark with BERTLARGE.
- (Mostly) the same hyperparameters for all datasets.
- Additional tricks:

$$\theta_\tau = \theta_{\text{pretrained}} + \delta_\tau$$

Initialized to zero.

$$\delta_\tau = z_\tau \odot w_\tau, \qquad z_\tau \in \{0,1\}^d, \ \boxed{w_\tau \in \mathbb{R}^d}$$

$$p(z_\tau; \boxed{\alpha_\tau}) = \prod_{i=1}^{d} \sigma(\alpha_{\tau,i})^{z_{\tau,i}} \times (1 - \sigma(\alpha_{\tau,i}))^{1 - z_{\tau,i}}$$

Initialized to positive value to discourage sparsity in the beginning.

Results

	Total params	New params per task	QNLI*	SST-2	MNLI_m	MNLI_{mm}	CoLA	MRPC	STS-B	RTE	QQP	Avg
Full finetuning	9.00×	100%	91.1	94.9	86.7	85.9	60.5	89.3	87.6	70.1	72.1	80.9

Results

Total number of parameters for all 9 tasks as a multiplier on top of BERT$_{LARGE}$

	Total params	New params per task	QNLI*	SST-2	MNLI$_m$	MNLI$_{mm}$	CoLA	MRPC	STS-B	RTE	QQP	Avg
Full finetuning	9.00×	100%	91.1	94.9	86.7	85.9	60.5	89.3	87.6	70.1	72.1	80.9

Results

Total number of parameters for all 9 tasks as a multiplier on top of BERTLARGE

	Total params	New params per task	QNLI*	SST-2	MNLI_m	MNLI_{mm}	CoLA	MRPC	STS-B	RTE	QQP	Avg
Full finetuning	9.00×	100%	91.1	94.9	86.7	85.9	60.5	89.3	87.6	70.1	72.1	80.9

Additional parameters per task (as a function of BERTLARGE)

Results

Total number of parameters for all 9 tasks as a multiplier on top of BERTLARGE

	Total params	New params per task	QNLI*	SST-2	MNLI_m	MNLI_{mm}	CoLA	MRPC	STS-B	RTE	QQP	Avg
Full finetuning	9.00×	100%	91.1	94.9	86.7	85.9	60.5	89.3	87.6	70.1	72.1	80.9

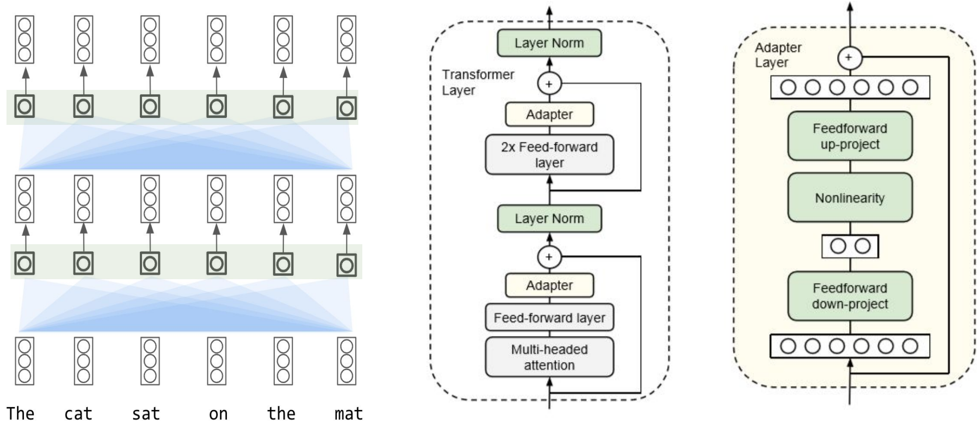
Additional parameters per task (as a function of BERTLARGE)

Average GLUE performance

Results

	Total params	New params per task	QNLI*	SST-2	MNLI_m	MNLI_{mm}	CoLA	MRPC	STS-B	RTE	QQP	Avg
Full finetuning	9.00×	100%	91.1	94.9	86.7	85.9	60.5	89.3	87.6	70.1	72.1	80.9
Adapters	1.32×	3.6%	90.7	94.0	84.9	85.1	59.5	89.5	86.9	71.5	71.8	80.4

Adapters from Houlsby et al. '19



Results

	Total params	New params per task	QNLI*	SST-2	MNLI_m	MNLI_{mm}	CoLA	MRPC	STS-B	RTE	QQP	Avg
Full finetuning	9.00×	100%	91.1	94.9	86.7	85.9	60.5	89.3	87.6	70.1	72.1	80.9
Adapters	1.32×	3.6%	90.7	94.0	84.9	85.1	59.5	89.5	86.9	71.5	71.8	80.4
Last layer	1.34×	3.8%	79.8	91.6	71.4	72.9	40.2	80.1	67.3	58.6	63.3	68.2
Non-adap. diff pruning	1.05×	0.5%	89.7	93.6	84.9	84.8	51.2	81.5	78.2	61.5	68.6	75.5

1. Fine-tune as usual to obtain task-specific parameters θ_τ
2. Calculate diff vector as $\theta_\tau - \theta_{\text{pretrained}}$
3. Magnitude pruning + fine-tuning on diff vector.

Results

	Total params	New params per task	QNLI*	SST-2	MNLI_m	MNLI_{mm}	CoLA	MRPC	STS-B	RTE	QQP	Avg
Full finetuning	9.00×	100%	91.1	94.9	86.7	85.9	60.5	89.3	87.6	70.1	72.1	80.9
Adapters	1.32×	3.6%	90.7	94.0	84.9	85.1	59.5	89.5	86.9	71.5	71.8	80.4
Last layer	1.34×	3.8%	79.8	91.6	71.4	72.9	40.2	80.1	67.3	58.6	63.3	68.2
Non-adap. diff pruning	1.05×	0.5%	89.7	93.6	84.9	84.8	51.2	81.5	78.2	61.5	68.6	75.5
Diff pruning	1.05×	0.5%	92.9	93.8	85.7	85.6	60.5	87.0	83.5	68.1	70.6	79.4
Diff pruning (struct.)	1.05×	0.5%	93.3	94.1	86.4	86.0	61.1	89.7	86.0	70.6	71.1	80.6

Memory-efficiency vs. Model Compression

(with BERTBASE)

	Total params	New params per task	QNLI	SST-2	MNLI_m	MNLI_{mm}	CoLA	MRPC	STS-B	RTE	QQP	Avg
Full finetuning	9.00×	100%	90.9	93.4	83.9	83.4	52.8	87.5	85.2	67.0	71.1	79.5
DistilBERT_6	5.53×	61.5%	88.9	92.5	82.6	81.3	49.0	86.9	81.3	58.4	70.1	76.8
TinyBERT_6	5.53×	61.5%	90.4	93.1	84.6	83.2	51.1	87.3	83.7	70.0	71.6	79.4
DistilBERT_4	4.31×	47.9%	85.2	91.4	78.9	78.0	32.8	82.4	76.1	54.1	68.5	71.9
TinyBERT_4	1.20×	13.3%	87.7	92.6	82.5	81.8	44.1	86.4	80.4	66.6	71.3	77.0
$\text{MobileBERT}_{\text{TINY}}$	1.24×	13.9%	89.5	91.7	81.5	81.6	46.7	87.9	80.1	65.1	68.9	77.0
Diff pruning (struct.)	1.05×	0.5%	90.0	92.9	83.7	83.4	52.0	88.0	84.5	66.4	70.3	79.0

Memory-efficiency vs. Model Compression

(with $\text{BERT}_{\text{BASE}}$)

	Total params	New params per task	QNLI	SST-2	MNLI_m	MNLI_{mm}	CoLA	MRPC	STS-B	RTE	QQP	Avg
Full finetuning	9.00×	100%	90.9	93.4	83.9	83.4	52.8	87.5	85.2	67.0	71.1	79.5
DistilBERT_6	5.53×	61.5%	88.9	92.5	82.6	81.3	49.0	86.9	81.3	58.4	70.1	76.8
TinyBERT_6	5.53×	61.5%	90.4	93.1	84.6	83.2	51.1	87.3	83.7	70.0	71.6	79.4
DistilBERT_4	4.31×	47.9%	85.2	91.4	78.9	78.0	32.8	82.4	76.1	54.1	68.5	71.9
TinyBERT_4	1.20×	13.3%	87.7	92.6	82.5	81.8	44.1	86.4	80.4	66.6	71.3	77.0
$\text{MobileBERT}_{\text{TINY}}$	1.24×	13.9%	89.5	91.7	81.5	81.6	46.7	87.9	80.1	65.1	68.9	77.0
Diff pruning (struct.)	1.05×	0.5%	90.0	92.9	83.7	83.4	52.0	88.0	84.5	66.4	70.3	79.0

Requires 120%-553% $\text{BERT}_{\text{BASE}}$ parameters for all 9 tasks.
⇒ Diff pruning becomes more memory-efficient as the number of tasks increases.

Memory-efficiency vs. Model Compression

Caveat: these models are smaller ⇒ faster inference.

(with BERTBASE)

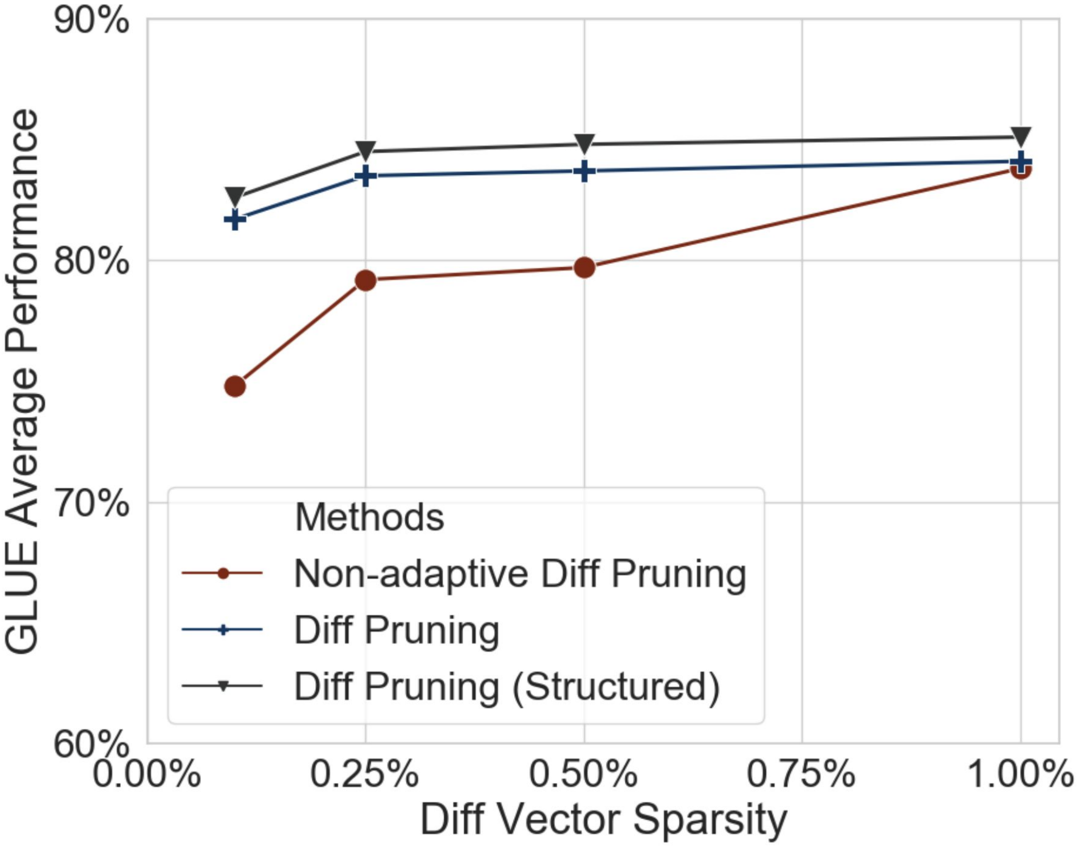
	Total params	New params per task	QNLI	SST-2	MNLI_m	MNLI_{mm}	CoLA	MRPC	STS-B	RTE	QQP	Avg
Full finetuning	9.00×	100%	90.9	93.4	83.9	83.4	52.8	87.5	85.2	67.0	71.1	79.5
DistilBERT_6	5.53×	61.5%	88.9	92.5	82.6	81.3	49.0	86.9	81.3	58.4	70.1	76.8
TinyBERT_6	5.53×	61.5%	90.4	93.1	84.6	83.2	51.1	87.3	83.7	70.0	71.6	79.4
DistilBERT_4	4.31×	47.9%	85.2	91.4	78.9	78.0	32.8	82.4	76.1	54.1	68.5	71.9
TinyBERT_4	1.20×	13.3%	87.7	92.6	82.5	81.8	44.1	86.4	80.4	66.6	71.3	77.0
$\text{MobileBERT}_{\text{TINY}}$	1.24×	13.9%	89.5	91.7	81.5	81.6	46.7	87.9	80.1	65.1	68.9	77.0
Diff pruning (struct.)	1.05×	0.5%	90.0	92.9	83.7	83.4	52.0	88.0	84.5	66.4	70.3	79.0

Requires 120%-553% BERTBASE parameters for all 9 tasks.
⇒ Diff pruning becomes more memory-efficient as the number of tasks increases.

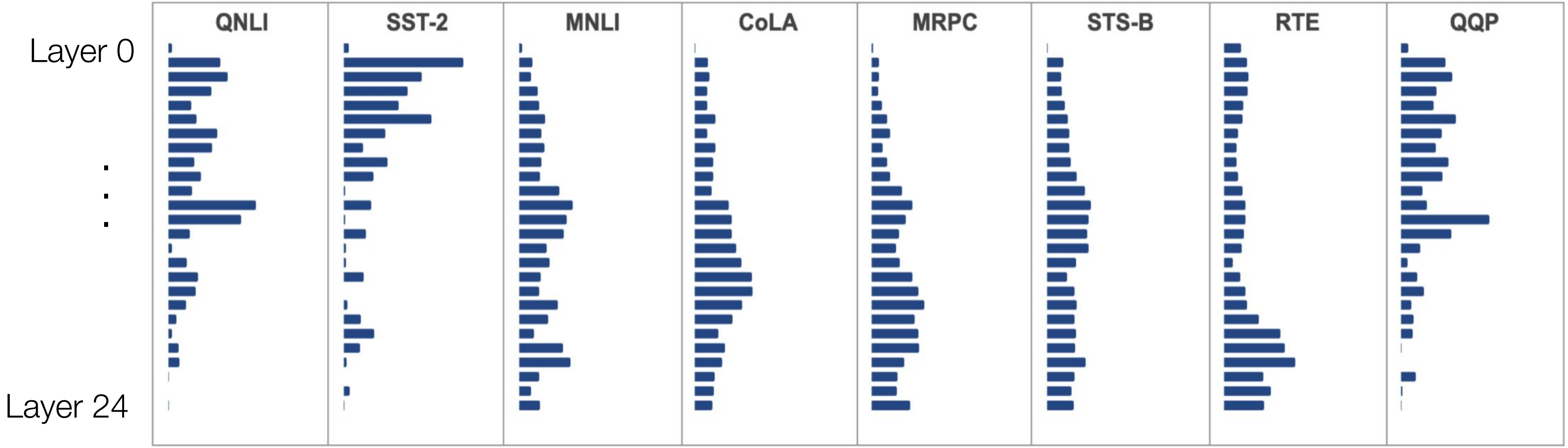
Memory-efficiency vs. Model Compression

	New params per task	Storage (MB) per task
Full finetuning	100%	1297.0
Adapters (weights only)	3.6%	49.0
Diff pruning (positions + weights)	0.5%	13.6

Analysis: Sparsity vs. Performance



Analysis: Distribution of Non-zero Diffs



Summary

- Open questions:
  - Is memory-scaling per task actually a concern?
  - Adapters vs. prefix-tuning vs. additive updates?
  - Sparse fine-tuning for continual learning?

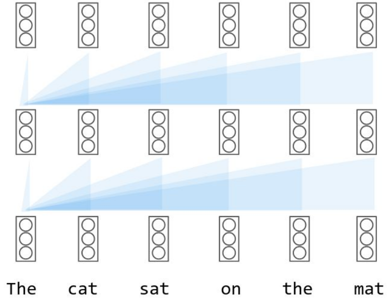
Summary

- Open questions:
    - Is memory-scaling per task actually a concern?
    - Adapters vs. prefix-tuning vs. additive updates?
    - Sparse fine-tuning for continual learning?
- Recent works:
    - BitFit [Ben-Zaken et al. '22]: Only tune bias vectors ⇒ competitive performance with only 0.08% parameters per task!
    - FISH [Sung et al. '21]: Use (an approximation of) Fisher Information matrix to prune diff vector.

Efficient Transfer Learning with Language Models

Memory Efficiency:
“Parameter-Efficient Transfer Learning with Diff Pruning”

(with Demi Guo, Alexander Rush; ACL ’21)



Inference Efficiency:
“Co-training Improves Prompt-based Learning for Large Language Models”



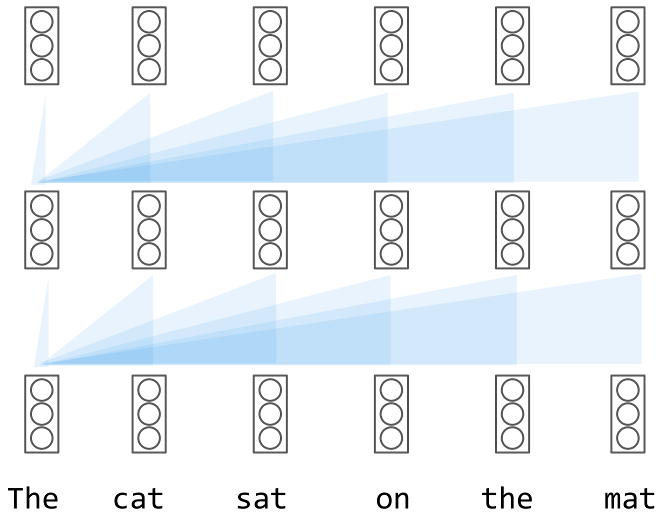
(with Hunter Lang, Monica Agrawal, David Sontag; ICML ’22)

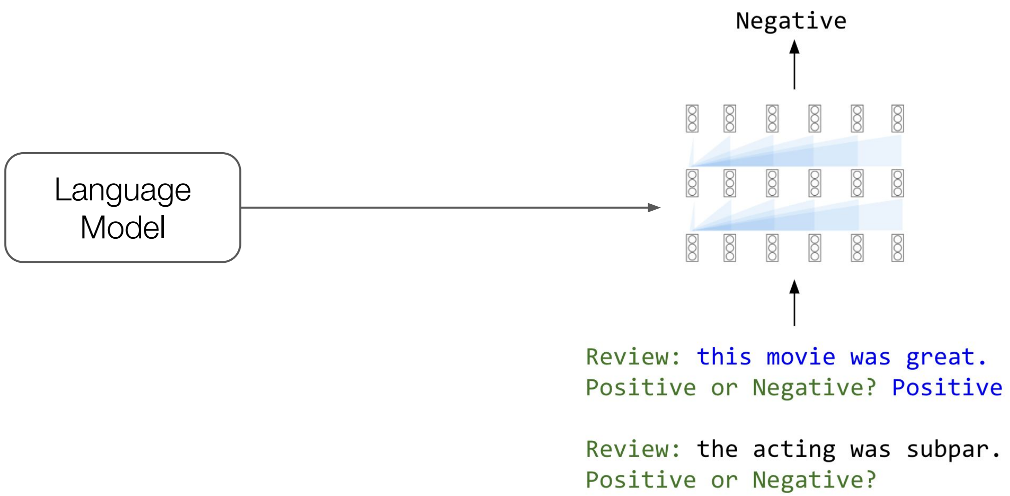
Transfer Learning via Prompting

Pretraining Phase

Conditioning via Language "Prompts"

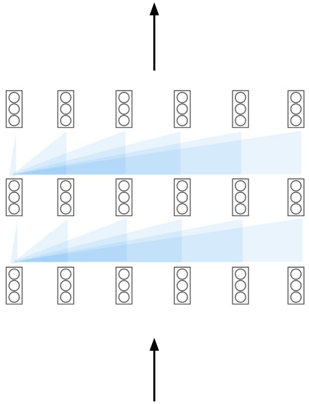
Task-Specific Model





Prompt-based Few- and Zero-shot Learning

```
Je ne suis pas un chat
```

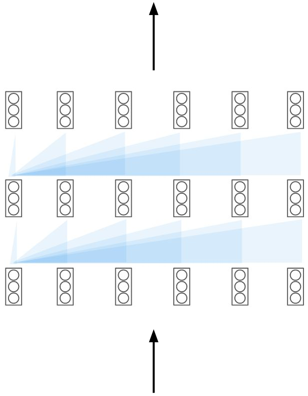


```
Translate the following sentence
from English to French.

English: I’m not a cat
French:
```

Zero-shot Learning for
Machine Translation

```
Negative
```



```
Review: this movie was great.
Positive or Negative? Positive

Review: the acting was subpar.
Positive or Negative?
```

Few-shot Learning for
Text Classification

Prompt-based Learning

✓ Model remains fixed ⇒ memory does not increase with the number of tasks (unlike BERT fine-tuning).

✓ Non-trivial performance with only a few (or no) examples.

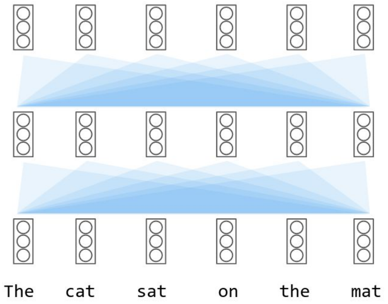
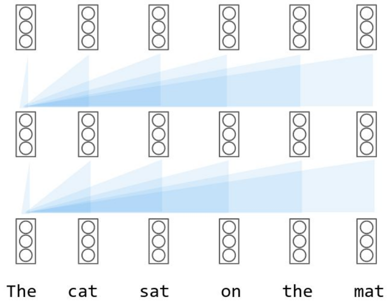
Prompt-based Learning

✓ Model remains fixed ⇒ memory does not increase with the number of tasks (unlike BERT fine-tuning).

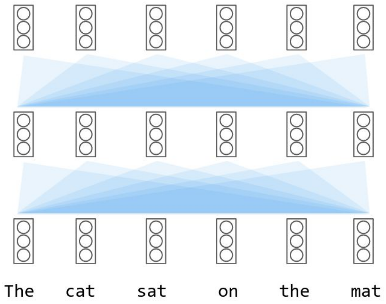
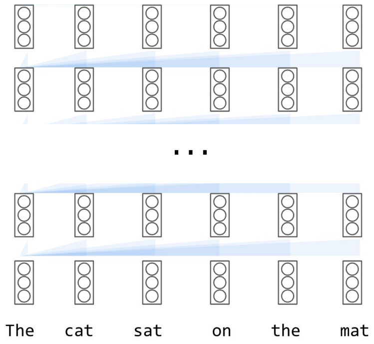
✓ Non-trivial performance with only a few (or no) examples.

✗ Prompt-based capabilities seem to emerge only when model sizes are large enough [Wei et al. '21] ⇒ inference is expensive!

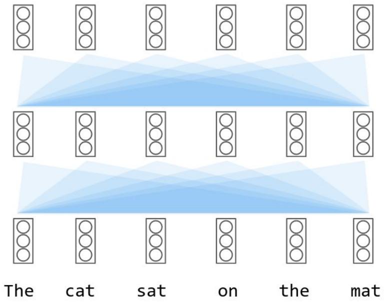
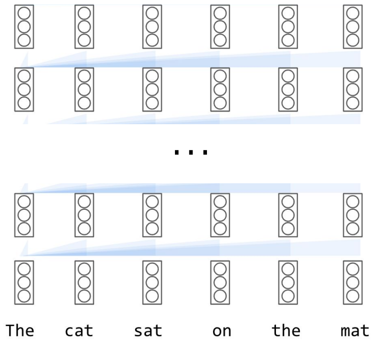
Inference Efficiency for Few-shot Prompting

	Examples	Size	Adaptation	Labeled data	Inference
	BERT RoBERTa XLNet BART T5	100M-10B	Fine-tuning	>16	Fast
	GPT-3 GLaM T0 FLAN PaLM	10B-500B	Prompting	<16	Slow

Inference Efficiency for Few-shot Prompting

	Examples	Size	Adaptation	Labeled data	Inference
	BERT RoBERTa XLNet BART T5	100M-10B	Fine-tuning	>16	Fast
	GPT-3 GLaM T0 FLAN PaLM	10B-500B	Prompting	<16	**Really slow**

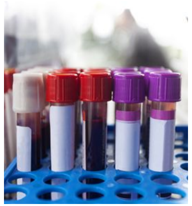
Inference Efficiency for Few-shot Prompting

	Examples	Size	Adaptation	Labeled data	Inference
	BERT RoBERTa XLNet BART T5	100M-10B	Fine-tuning	>16	**Fast**
	GPT-3 GLaM T0 FLAN PaLM	10B-500B	Prompting	**<16**	Really slow

Can we get the best of both worlds?

Co-Training [Blum and Mitchell '98]

- A semi-supervised approach for leveraging unlabeled data.
- Pair of models are trained over different "views" of the same underlying data.

View	$\phi_0(X)$	$\phi_1(X)$
Model	h_0	h_1
		
	Lab tests	X-ray

Co-Training [Blum and Mitchell ’98]

- A semi-supervised approach for leveraging unlabeled data.
- Pair of models are trained over different “views” of the same underlying data.

View	$\phi_0(X)$	$\phi_1(X)$
Model	h_0	h_1


Samoyed Dogs Are Basically A Breed Of Big, Fluffy, Sentient Clouds

Amy Pilkington
June 26, 2020

You know what I love? Dogs and doggo lingo.

Dogs, doggos, puppers, pupperinos, floofs, and woofers. Regardless of breed, a
good dog just puts a smile on my face, because all dogs are good dogs.

However, some breeds have the distinction of being their own category within the
lingo, and the samoyed is one such breed.

Text on web page

Query that led to article

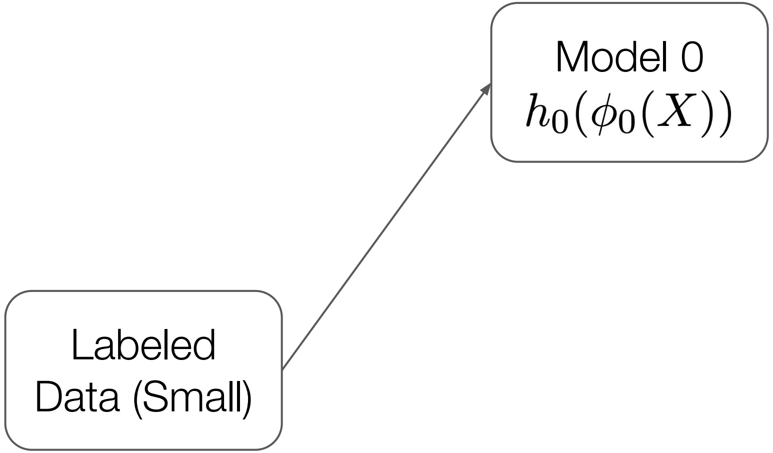
Co-Training [Blum and Mitchell '98]

- A semi-supervised approach for leveraging unlabeled data.
- Pair of models are trained over different "views" of the same underlying data.

View	$\phi_0(X)$	$\phi_1(X)$
Model	h_0	h_1

- The two models $h_0(\phi_0(X))$ and $h_1(\phi_1(X))$ are iteratively trained on confidently-labeled data points from the other model.

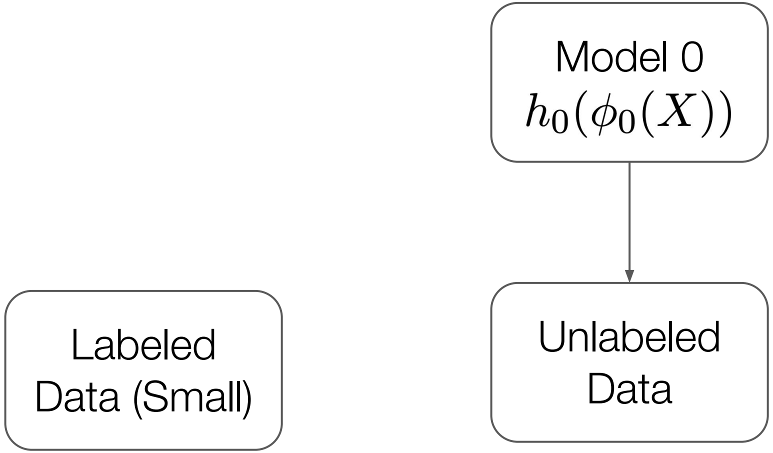
Co-Training



Round 1

- Train h_0 on small labeled data.

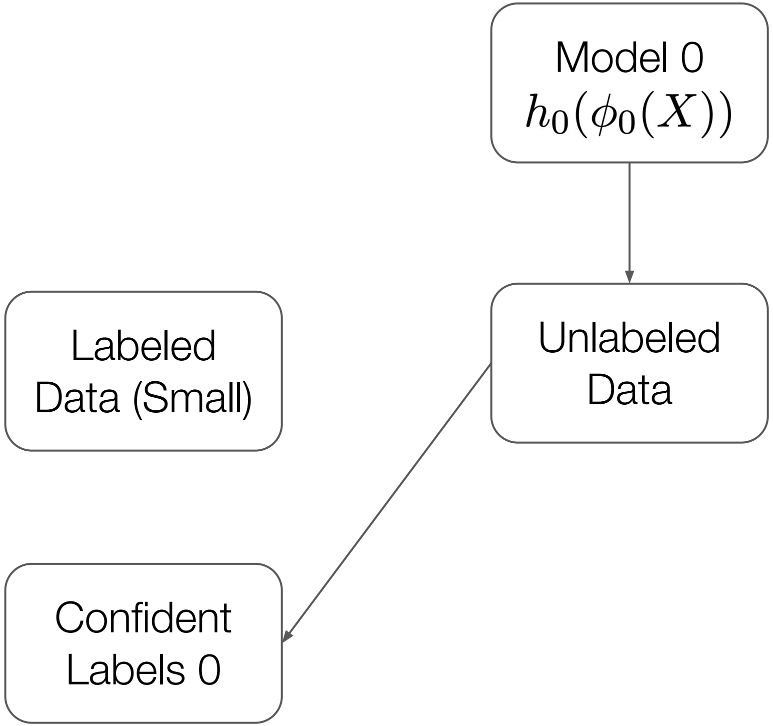
Co-Training



Round 1

- Train h_0 on small labeled data
- Apply h_0 on view $\phi_0(X)$ of unlabeled data.

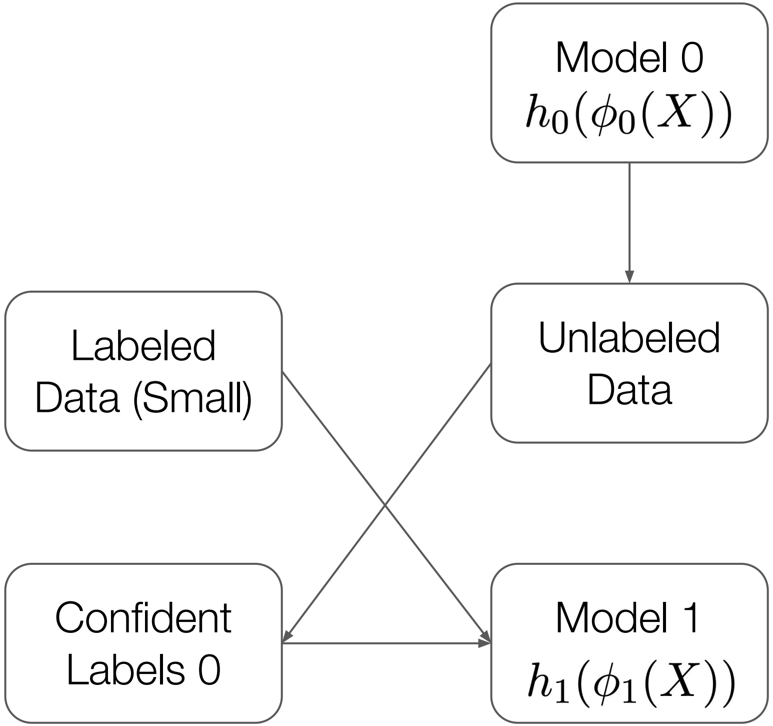
Co-Training



Round 1

- Train h_0 on small labeled data.
- Apply h_0 on view $\phi_0(X)$ of unlabeled data.
- Get confidently-labeled data as pseudo-labels.

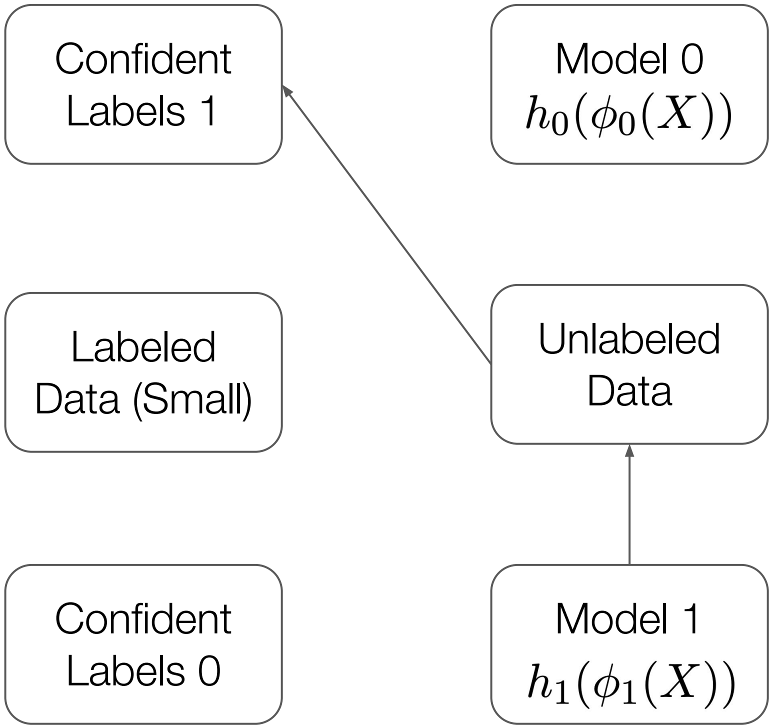
Co-Training



Round 1

- Train h_0 on small labeled data.
- Apply h_0 on view $\phi_0(X)$ of unlabeled data.
- Get confidently-labeled data as pseudo-labels.
- Train h_1 on view $\phi_1(X)$ on pseudo-labeled data.

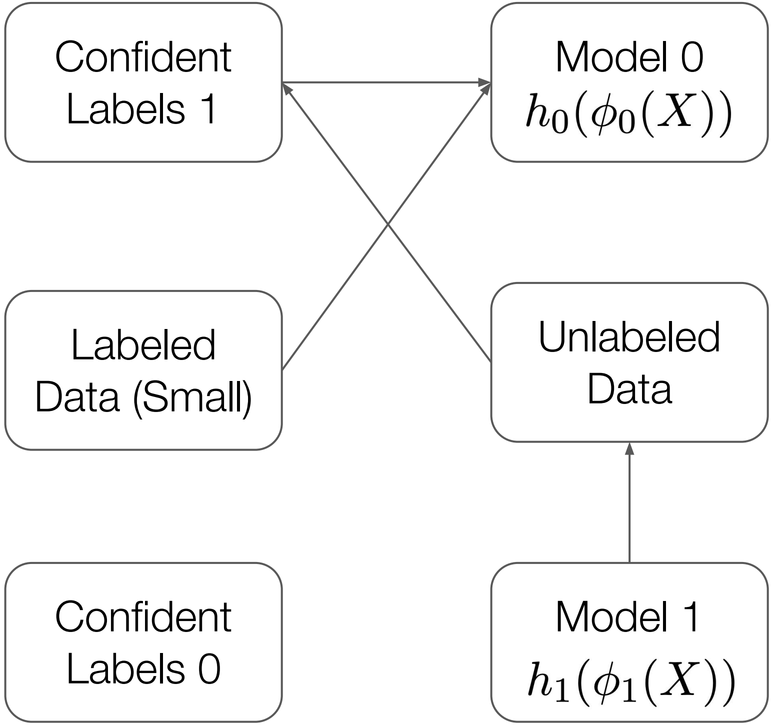
Co-Training



Round 1

- Apply h_1 on view $\phi_1(X)$ of unlabeled data.
- Get confidently-labeled data as pseudo-labels.

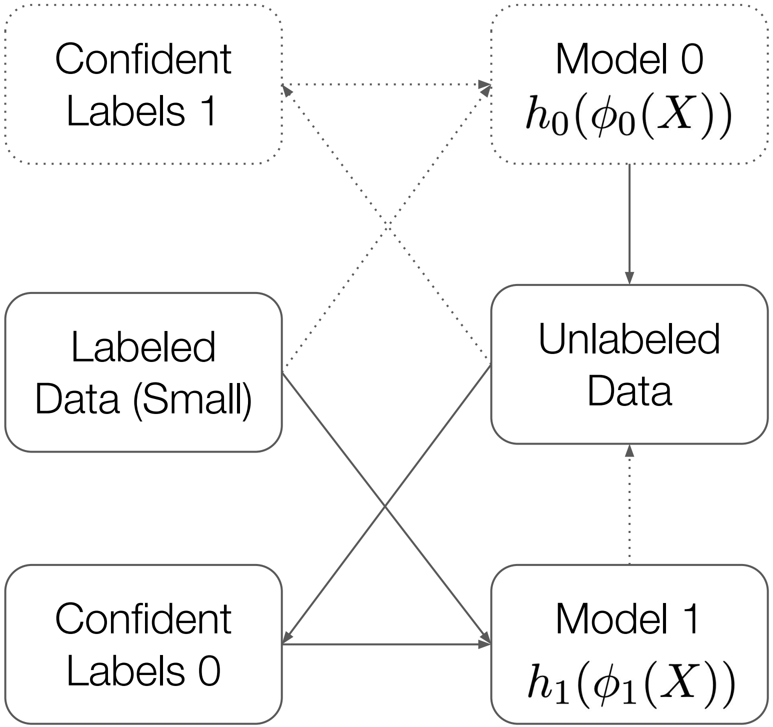
Co-Training



Round 1

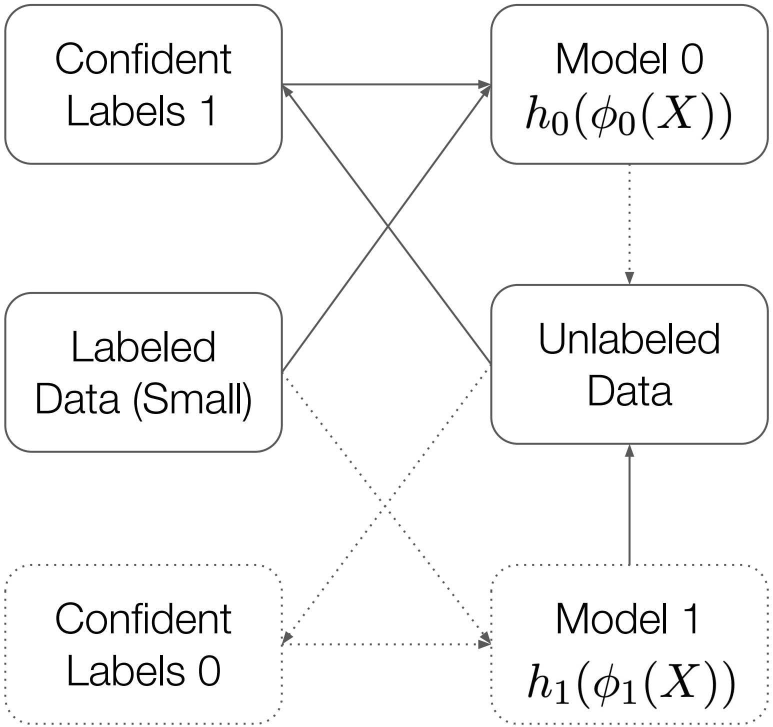
- Apply h_1 on view $\phi_1(X)$ of unlabeled data.
- Get confidently-labeled data as pseudo-labels.
- Retrain h_0 on view $\phi_0(X)$ on pseudo-labels.

Co-Training



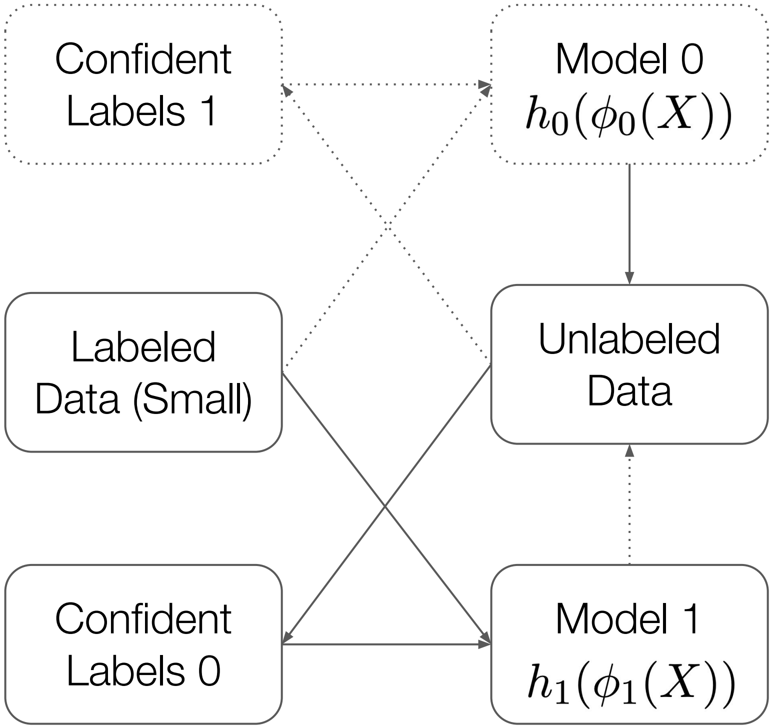
Round 2

Co-Training



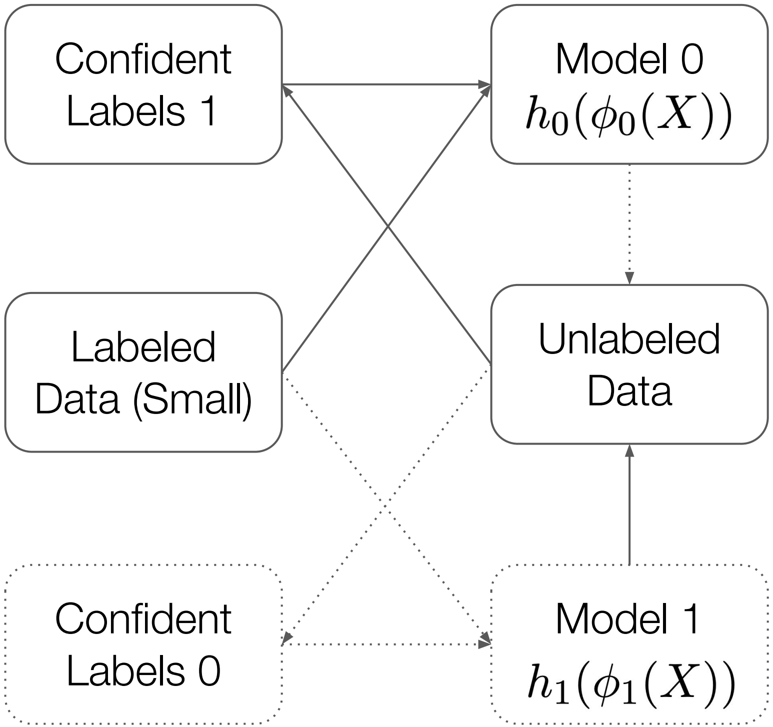
Round 2

Co-Training



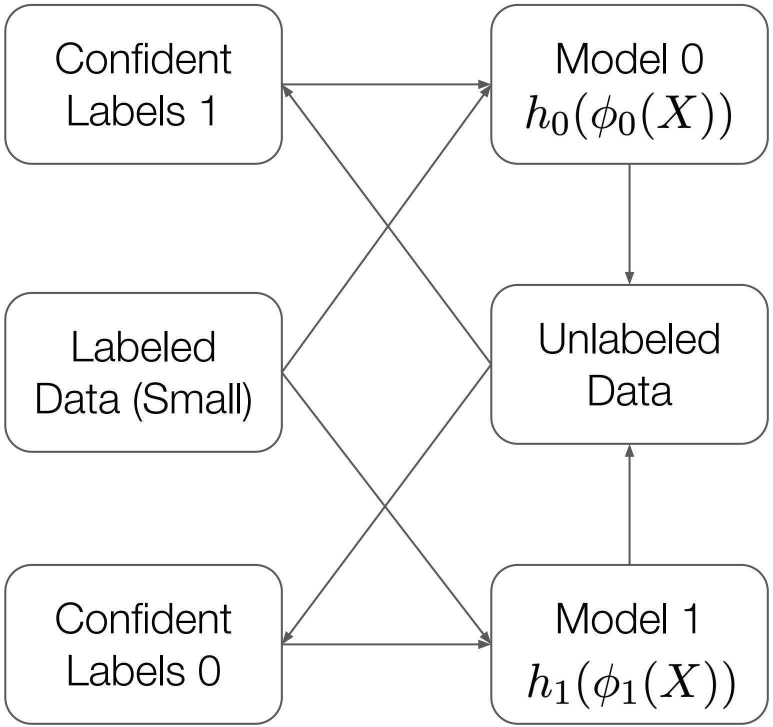
Round 3

Co-Training



Round 3

Co-Training

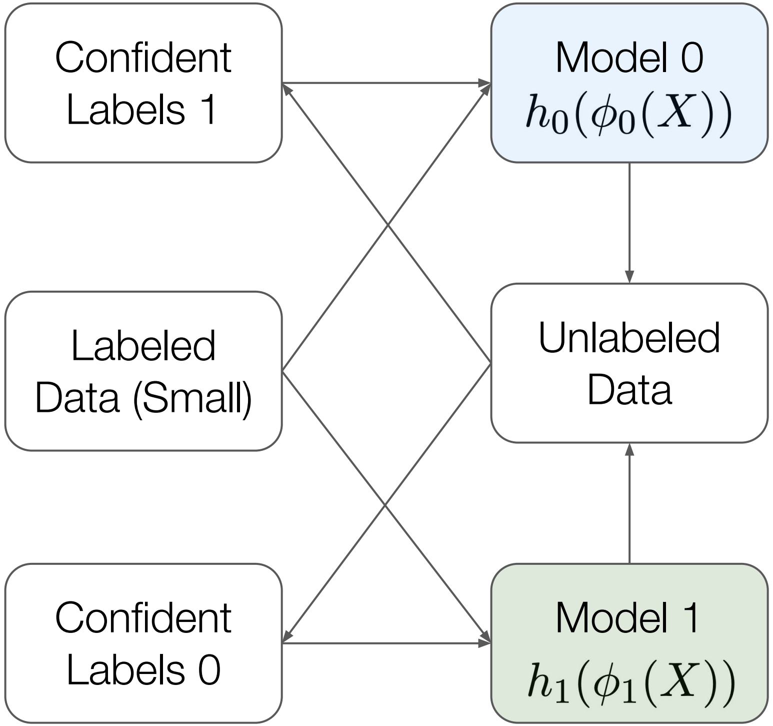


If the views are “different enough”, then the learned classifier will have low error [Blum and Mitchel ’98; Balcan et al. ’05]

$\phi_0(X)$ — Pretrained LM

$\phi_1(X)$ — Another pretrained LM with different inductive biases?

Co-Training for Inference Efficiency

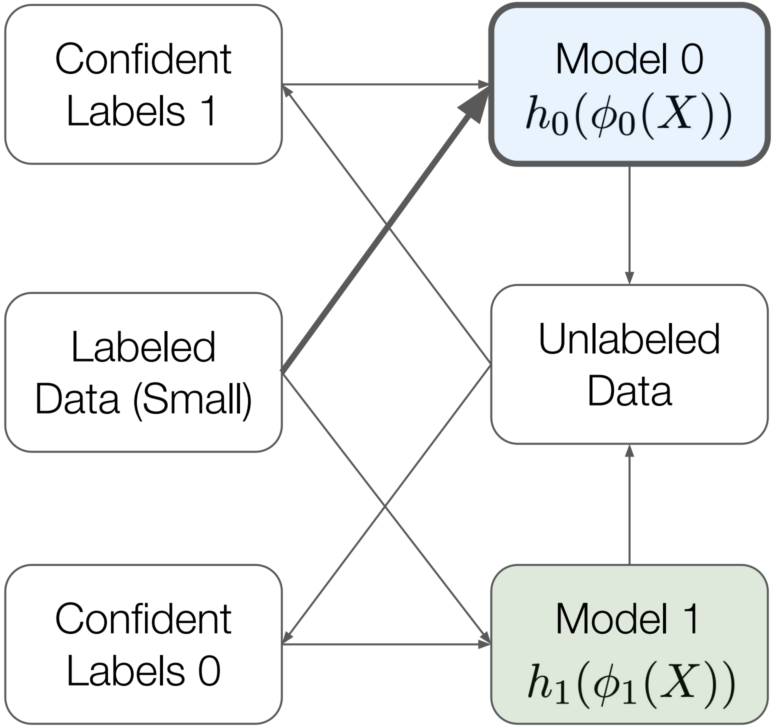


Final model

Simple idea:

- Prompted GPT-3 as the initial model.
- BERT as the other model ⇒ Faster inference!
- Implicit ensembling of different inductive biases.

Co-Training for Inference Efficiency



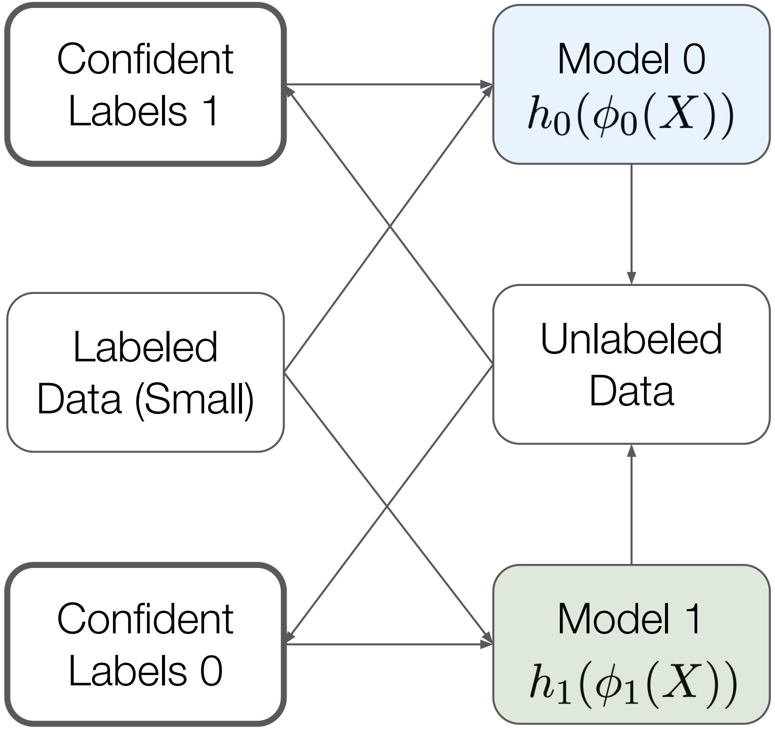
Simple idea:

- Prompted GPT-3 as the initial model.
- BERT as the other model ⇒ Faster inference!
- Implicit ensembling of different inductive biases.

Questions:

- How to learn a model over prompted GPT-3 to make it amenable to updating?

Co-Training for Inference Efficiency



Simple idea:

- Prompted GPT-3 as the initial model.
- BERT as the other model ⇒ Faster inference!
- Implicit ensembling of different inductive biases.

Questions:

- How to learn a model over prompted GPT-3 to make it amenable to updating?
- How to select confident labels?

Prompt-based Few-shot Learning

Example: RTE (Textual Entailment) with two labeled examples (k=2)

Usual approach: k-shot prompting ⇒ Feed k labeled data as a single prompt

Prompt-based Few-shot Learning

Example: RTE (Textual Entailment) with two labeled examples (k=2)

Usual approach: k-shot prompting ⇒ Feed k labeled data as a single prompt

```
Oil prices fall back as Yukos oil threat lifted.
Question: Oil prices dropped. True, False, or Unknown?
Answer: True

The cost of consumer of the United States fell in June.
Question: U.S. consumer spending dived in June. True, False,
or Unknown?
Answer: False

Hepburn's family will receive proceeds from the sale.
Question: Proceeds go to Hepburn's family. True, False or
Unknown?
```

Prompt Labeled examples Unlabeled input

Prompt-based Few-shot Learning

Example: RTE (Textual Entailment) with two labeled examples (k=2)

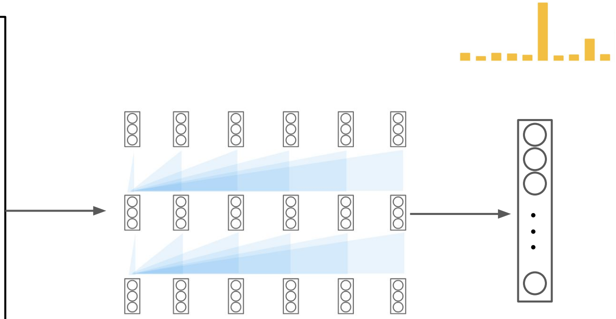
Usual approach: k-shot prompting ⇒ Feed k labeled data as a single prompt

```
Oil prices fall back as Yukos oil threat lifted.
Question: Oil prices dropped. True, False, or Unknown?
Answer: True

The cost of consumer of the United States fell in June.
Question: U.S. consumer spending dived in June. True, False,
or Unknown?
Answer: False

Hepburn's family will receive proceeds from the sale.
Question: Proceeds go to Hepburn's family. True, False or
Unknown?
```

Prompt   Labeled examples   Unlabeled input



Distribution over label tokens (i.e., verbalizer)

{True, False, Unknown, true, false, unknown, Yes, No, yes, no, …}

Prompt-based Few-shot Learning

Example: RTE (Textual Entailment) with two labeled examples (k=2)

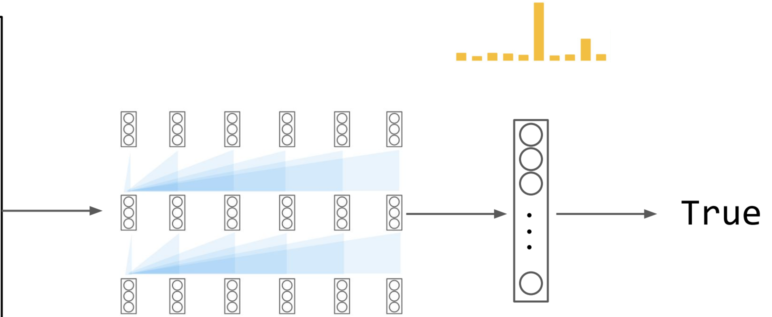
Usual approach: k-shot prompting ⇒ Feed k labeled data as a single prompt

```
Oil prices fall back as Yukos oil threat lifted.
Question: Oil prices dropped. True, False, or Unknown?
Answer: True

The cost of consumer of the United States fell in June.
Question: U.S. consumer spending dived in June. True, False,
or Unknown?
Answer: False

Hepburn's family will receive proceeds from the sale.
Question: Proceeds go to Hepburn's family. True, False or
Unknown?
```

Prompt   Labeled examples   Unlabeled input



Distribution over label tokens (i.e., verbalizer)

{True, False, Unknown, true, false, unknown, Yes, No, yes, no, …}

$\phi_0(X)$: Prompted GPT-3 probabilities as view 0

Example: RTE (Textual Entailment) with two labeled examples (k=2)

Our approach: k one-shot prompts ⇒ Concatenate GPT-3 output probabilities from k prompted models

$\phi_0(X)$: Prompted GPT-3 probabilities as view 0

Example: RTE (Textual Entailment) with two labeled examples (k=2)

Our approach: k one-shot prompts ⇒ Concatenate GPT-3 output probabilities from k prompted models

```
Oil prices fall back as Yukos oil threat lifted.
Question: Oil prices dropped. True, False, or Unknown?
Answer: True

Hepburn's family will receive proceeds from the sale.
Question: Proceeds go to Hepburn's family. True, False or
Unknown?
```

```
The cost of consumer of the United States fell in June.
Question: U.S. consumer spending dived in June. True,
False, or Unknown?
Answer: False

Hepburn's family will receive proceeds from the sale.
Question: Proceeds go to Hepburn's family. True, False or
Unknown?
```

Prompt   Labeled examples   Unlabeled input

$\phi_0(X)$: Prompted GPT-3 probabilities as view 0

Example: RTE (Textual Entailment) with two labeled examples (k=2)

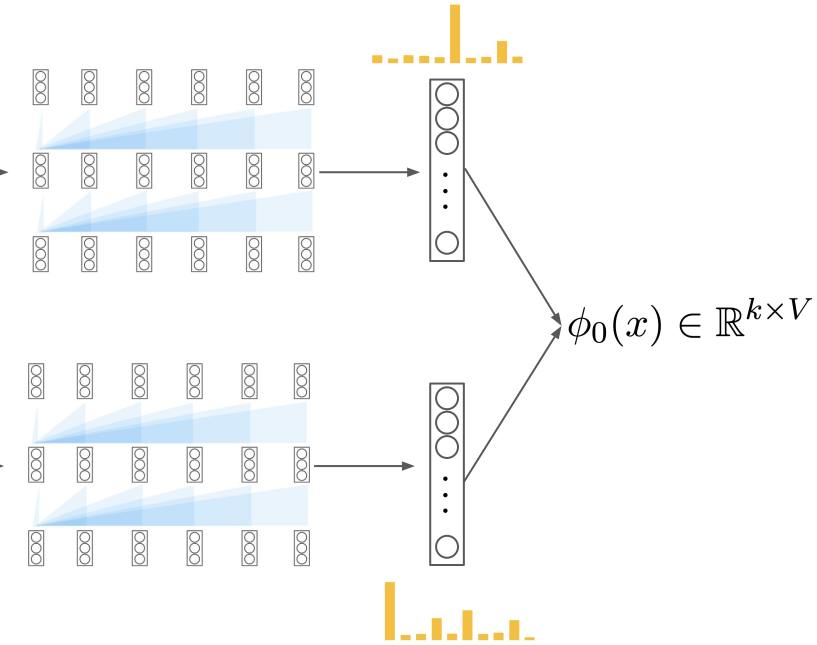
Our approach: k one-shot prompts ⇒ Concatenate GPT-3 output probabilities from k prompted models

```
Oil prices fall back as Yukos oil threat lifted.
Question: Oil prices dropped. True, False, or Unknown?
Answer: True

Hepburn’s family will receive proceeds from the sale.
Question: Proceeds go to Hepburn’s family. True, False or
Unknown?
```

```
The cost of consumer of the United States fell in June.
Question: U.S. consumer spending dived in June. True,
False, or Unknown?
Answer: False

Hepburn’s family will receive proceeds from the sale.
Question: Proceeds go to Hepburn’s family. True, False or
Unknown?
```



Prompt   Labeled examples   Unlabeled input

h_0 : Label model for aggregating GPT-3 outputs

- Simple averaging does not work well because (i) the probabilities are not well calibrated [Zhao et al. '21], (ii) there are no learnable parameters.

$$h_0(\phi_0(x)) = \frac{1}{k}\sum_{i=1}^{k}\phi_0^{(i)}(x)$$

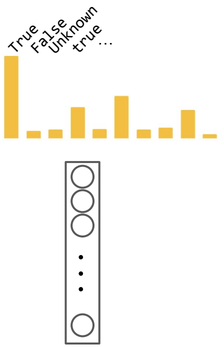
- Parameterized label model over $\phi_0(x) \in \mathbb{R}^{k\times V}$:

$$\mathbf{l}_i = \mathrm{ReLU}\left(W^{(i)}\phi_0^{(i)}(x)\right) \qquad W^{(i)} \in \mathbb{R}^{l\times V}$$

$$h_0(x; W, \alpha) = \mathrm{softmax}\left(\sum_{i=1}^{k}\alpha_i \mathbf{l}_i\right)$$

h_0 : Label model for aggregating GPT-3 outputs

$$\mathbf{l}_i = \mathrm{ReLU}\left(W^{(i)}\phi_0^{(i)}(x)\right)$$



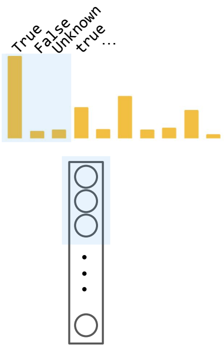
$$\phi^{(i)}(x) \in \mathbb{R}^V$$

l Label tokens
`{True, False, Unknown}`

V Verbalizer tokens
`{True, False, Unknown, true, false, unknown, Yes, No, yes, no, ...}`

h_0 : Label model for aggregating GPT-3 outputs

$$\mathbf{l}_i = \mathrm{ReLU}\left(W^{(i)}\phi_0^{(i)}(x)\right)$$



$\phi^{(i)}(x) \in \mathbb{R}^V$

l Label tokens
`{True, False, Unknown}`

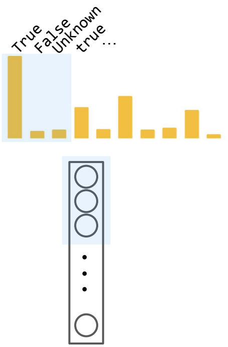
V Verbalizer tokens
`{True, False, Unknown, true, false, unknown, Yes, No, yes, no, …}`

Assume WLOG that the first l dimensions of $\phi^{(i)}(x)$ correspond to label tokens.

(See paper on how to obtain the set of verbalizer tokens in a task-agnostic way)

h_0 : Label model for aggregating GPT-3 outputs

$$\mathbf{l}_i = \mathrm{ReLU}\left(W^{(i)}\phi_0^{(i)}(x)\right)$$



$$\begin{bmatrix} \mathtt{a} & \mathtt{0} & \mathtt{0} & \mathtt{0} & \mathtt{0} & \dots & \mathtt{0} \\ \mathtt{0} & \mathtt{b} & \mathtt{0} & \mathtt{0} & \mathtt{0} & \dots & \mathtt{0} \\ \mathtt{0} & \mathtt{0} & \mathtt{c} & \mathtt{0} & \mathtt{0} & \dots & \mathtt{0} \end{bmatrix}$$

$$W^{(i)} \in \mathbb{R}^{l \times V} \quad \phi^{(i)}(x) \in \mathbb{R}^{V}$$

$$\mathsf{a} = \frac{1}{P_{\text{GPT-3}}(\textit{next word} = \texttt{True} \mid \textit{prompt} = \text{``''})}$$

$$\mathsf{b} = \frac{1}{P_{\text{GPT-3}}(\textit{next word} = \texttt{False} \mid \textit{prompt} = \text{``''})}$$

l Label tokens
`{True, False, Unknown}`

V Verbalizer tokens
`{True, False, Unknown, true, false, unknown, Yes, No, yes, no, …}`

Assume WLOG that the first l dimensions of $\phi^{(i)}(x)$ correspond to label tokens.

Part of the matrix $W^{(i)}$ applied to these tokens is initialized to $\mathrm{Diag}\left(\frac{\mathbf{1}}{\phi_0^{(i)}(x_{cf})}\right)$

where $\phi_0^{(i)}(x_{cf})$ is label probability vector the output from an empty prompt [Zhao et al. '21].

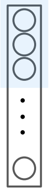
(Rest are initialized to 0.)

h_0 : Label model for aggregating GPT-3 outputs

$$\mathbf{l}_i = \mathrm{ReLU}\left(W^{(i)}\phi_0^{(i)}(x)\right)$$

Both `True` and `true` would contribute to `True`

$$\begin{bmatrix} 4 & 0 & 0 & 0 & 2 & \dots & 0 \\ 0 & 3 & 0 & 0 & 0 & \dots & 0 \\ 0 & 0 & 2 & 0 & 0 & \dots & 0 \end{bmatrix}$$



$$W^{(i)} \in \mathbb{R}^{l \times V} \quad \phi^{(i)}(x) \in \mathbb{R}^{V}$$

$$\mathrm{a} = \frac{1}{P_{\text{GPT-3}}(\textit{next word} = \texttt{True} \mid \textit{prompt} = \text{``''})}$$

$$\mathrm{b} = \frac{1}{P_{\text{GPT-3}}(\textit{next word} = \texttt{False} \mid \textit{prompt} = \text{``''})}$$

l Label tokens
`{True, False, Unknown}`

V Verbalizer tokens
`{True, False, Unknown, true, false, unknown, Yes, No, yes, no, ...}`

Assume WLOG that the first l dimensions of $\phi^{(i)}(x)$ correspond to label tokens.

Part of the matrix $W^{(i)}$ applied to these tokens is initialized to

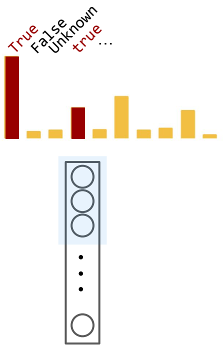
$$\mathrm{Diag}\left(\frac{\mathbf{1}}{\phi_0^{(i)}(x_{cf})}\right)$$

Inituition: initially the model uses the label token probabilities, but can learn to use verbalizer tokens that are related.

h_0 : Label model for aggregating GPT-3 outputs

$$\mathbf{l}_i = \mathrm{ReLU}\left(W^{(i)}\phi_0^{(i)}(x)\right)$$

Both `True` and `true` would contribute to `True`



$$\begin{bmatrix} 4 & 0 & 0 & 0 & 2 & \dots & 0 \\ 0 & 3 & 0 & 0 & 0 & \dots & 0 \\ 0 & 0 & 2 & 0 & 0 & \dots & 0 \end{bmatrix}$$

$$W^{(i)} \in \mathbb{R}^{l \times V} \quad \phi^{(i)}(x) \in \mathbb{R}^{V}$$

$$a = \frac{1}{P_{\text{GPT-3}}(\textit{next word} = \texttt{True} \mid \textit{prompt} = \text{“”})}$$

$$b = \frac{1}{P_{\text{GPT-3}}(\textit{next word} = \texttt{False} \mid \textit{prompt} = \text{“”})}$$

l Label tokens
`{True, False, Unknown}`

V Verbalizer tokens
`{True, False, Unknown, true, false, unknown, Yes, No, yes, no, ...}`

Assume WLOG that the first l dimensions of $\phi^{(i)}(x)$ correspond to label tokens.

Part of the matrix $W^{(i)}$ applied to these tokens is initialized to

$$\mathrm{Diag}\left(\frac{\mathbf{1}}{\phi_0^{(i)}(x_{cf})}\right)$$

Inituition: initially the model uses the label token probabilities, but can learn to use verbalizer tokens that are related.

ReLU can ignore certain prompt/label combinations.

h_0 : Label model for aggregating GPT-3 outputs

$$\mathbf{l}_i = \mathrm{ReLU}\left(W^{(i)}\phi_0^{(i)}(x)\right)$$

Aggregation layer that sums of probabilities from different verbalizer tokens into the label token.

$$h_0(x; W, \alpha) = \mathrm{softmax}\left(\sum_{i=1}^{k} \alpha_i \mathbf{l}_i\right)$$

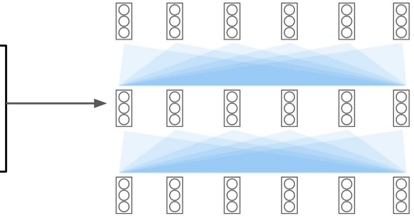
Calibration layer that learns to weight the different $\mathbf{l}_i \in \mathbb{R}^l$ vectors

The weights α_i are initialized to 1 to weight all prompts equally.

Final softmax over l labels gives probabilities with which to select confident labels. (Pseudo-labels to train the smaller model).

$\phi_1(X)$: Frozen embeddings from smaller MLM

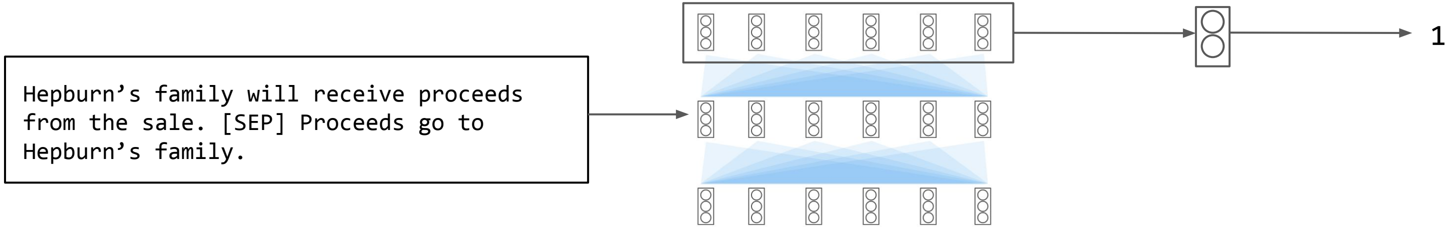
```
Hepburn’s family will receive proceeds
from the sale. [SEP] Proceeds go to
Hepburn’s family.
```



$\phi_1(X)$

DeBERTa
embeddings

h_1 : Classifier over MLM embeddings



$\phi_1(X)$

DeBERTa embeddings

h_1

MLP over embeddings

Pseudo-labeling

- Select β = 50% of unlabeled dataset initially.
- Increase this by β' = 10% at each round for 5 rounds of co-training.

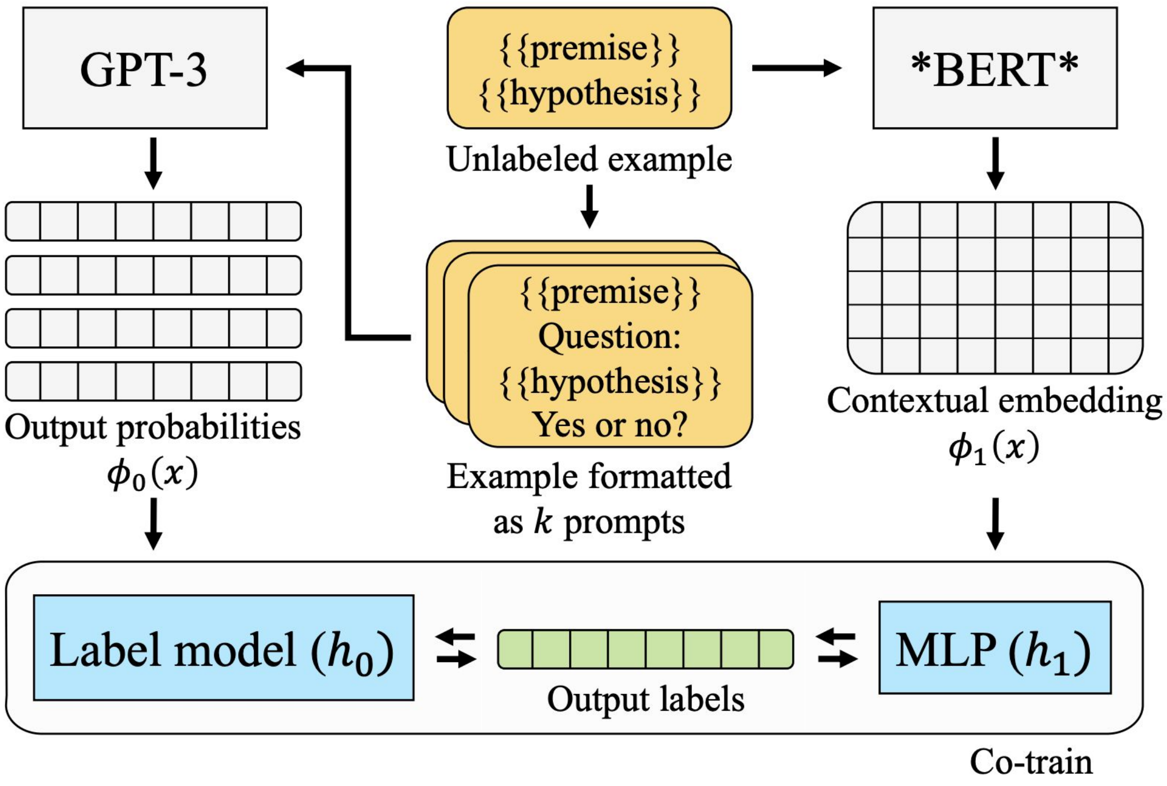
Pseudo-labeling

- Select β = 50% of unlabeled dataset initially.
- Increase this by β' = 10% at each round for 5 rounds of co-training.
- Initial model (from prompted GPT-3) can have very low probability for some labels ⇒ naive strategy of just taking the most confident labels can miss some labels.
- Make the (weak) assumption that each label is at least 1% of the dataset ⇒ ensures each label is included in each pseudo-labeling round.

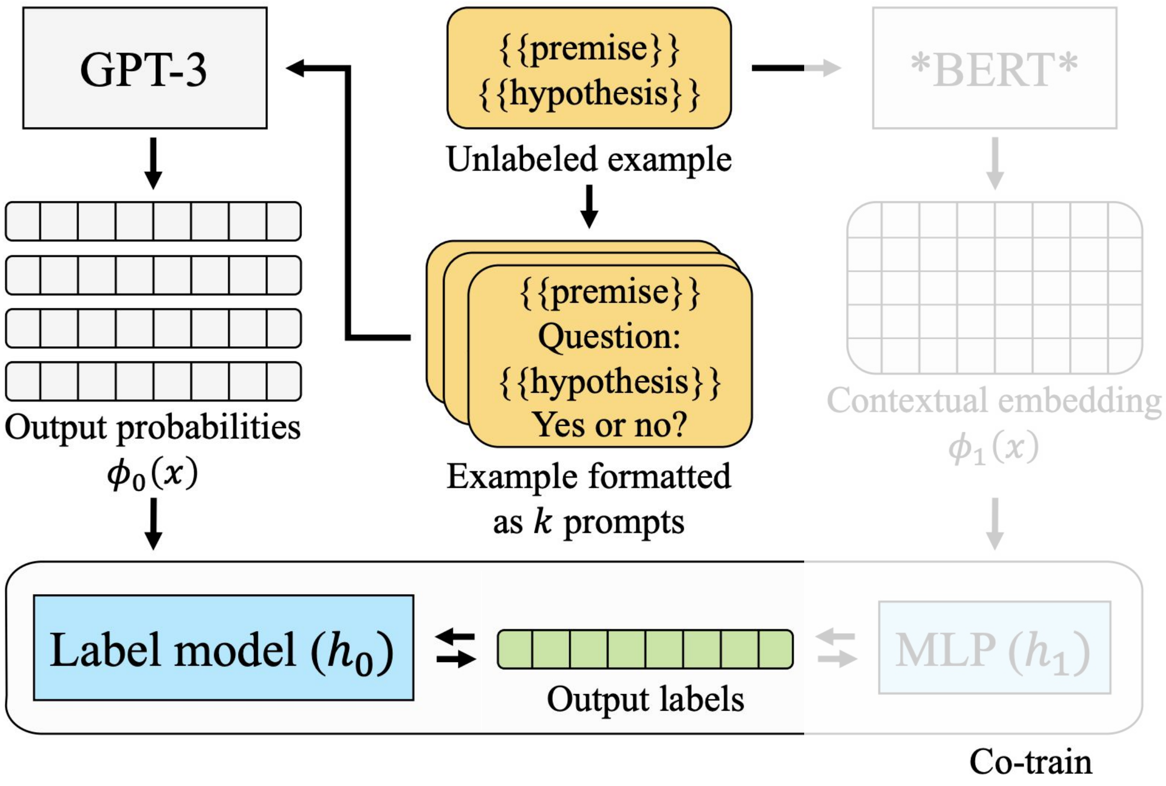
Pseudo-labeling

- Select β = 50% of unlabeled dataset initially.
- Increase this by β' = 10% at each round for 5 rounds of co-training.
- Initial model (from prompted GPT-3) can have very low probability for some labels ⇒ naive strategy of just taking the most confident labels can miss some labels.
- Make the (weak) assumption that each label is at least 1% of the dataset ⇒ ensures each label is included in each pseudo-labeling round.
- $\phi_0(X)$: use model confidence to select most confident labels
  $\phi_1(X)$: use *cut statistic* [Muhlenbach et al. '04] to select most confident labels to better take into account representation geometry.

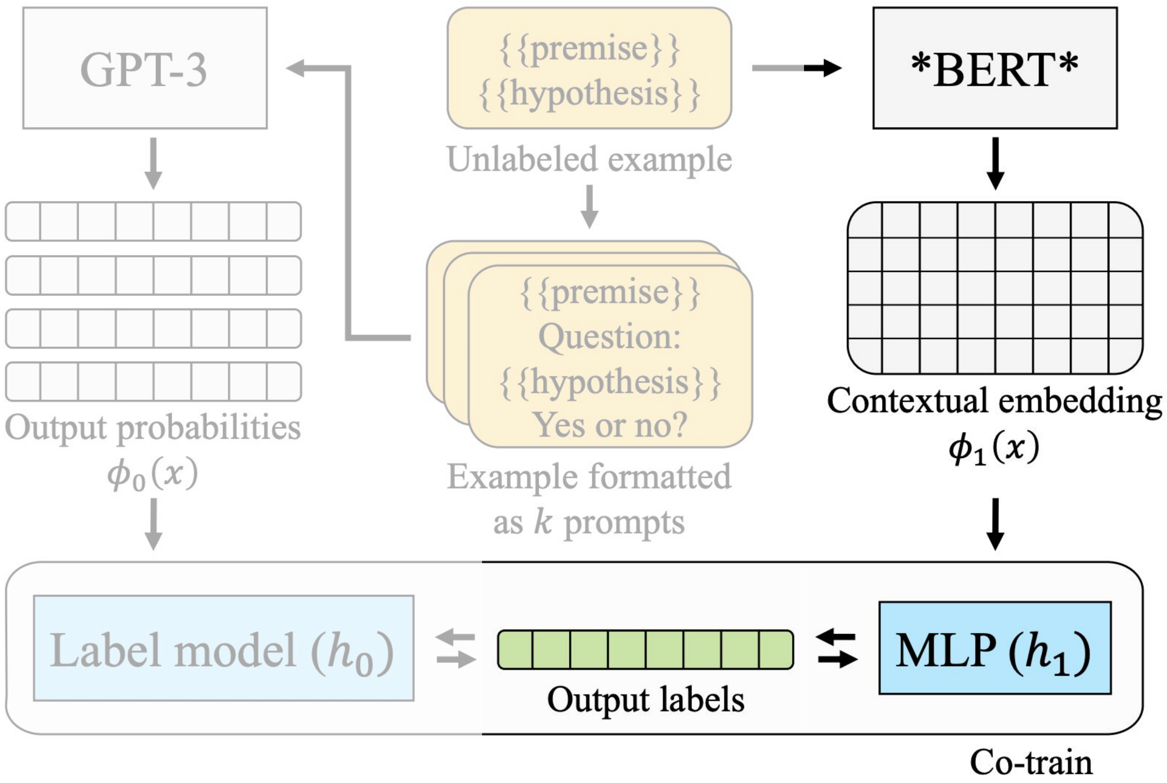
Putting it all together



Putting it all together



Putting it all together



Experiments

- Test on datasets traditionally difficult for few-shot learning:
  - Textual entailment (RTE, CB)
  - Question classification (TREC)

- Prompts/hyperparameters inherited from previous work to minimize label leakage.

- Co-training parameters (e.g., initial coverage, number of rounds) selected on small subset of TREC ⇒ TREC results not “true” few-shot.

- Same exact setup across all datasets.

Results: Few-shot

Using 4 labeled examples only

Model	View	RTE (2-class)	CB (3-class)	TREC (6-class)
GPT-3 4-shot (from Zhao et al. (2021))	*	58.7 (11.9)	45.2 (19.4)	60.2 (7.6)

Results: Few-shot

Using 4 labeled examples only

Model	View	RTE (2-class)	CB (3-class)	TREC (6-class)
GPT-3 4-shot (from Zhao et al. (2021))	*	58.7 (11.9)	45.2 (19.4)	60.2 (7.6)
Calibrate Before Use (CBU) (Zhao et al., 2021)	*	60.4 (8.1)	60.7 (6.7)	69.7 (1.4)

CBU [Zhao et al '21]: rescale GPT-3 probabilities based on null prompt

$$\mathrm{Diag}\left(\frac{\mathbf{1}}{\phi_0^{(i)}(x_{cf})}\right)$$

$$\mathsf{a} = \frac{1}{P_{\textsf{GPT-3}}(\textit{next word} = \texttt{True} \mid \textit{prompt} = \text{``''})}$$

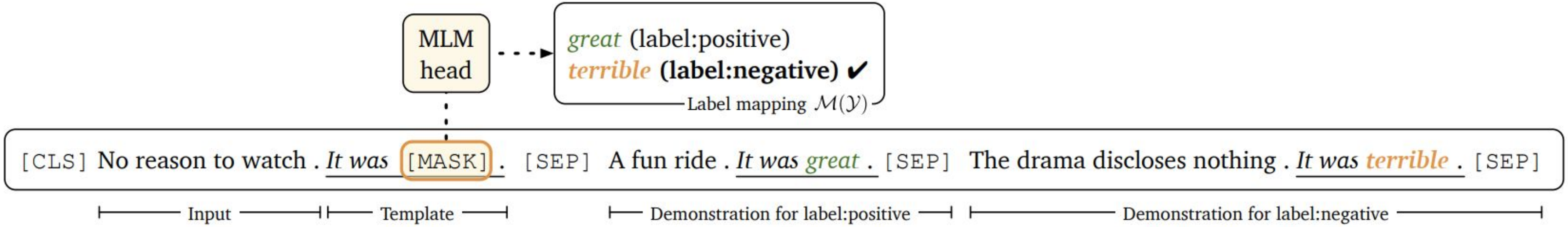
$$\mathsf{b} = \frac{1}{P_{\textsf{GPT-3}}(\textit{next word} = \texttt{False} \mid \textit{prompt} = \text{``''})}$$

Results: Few-shot

Using 4 labeled examples only

Model	View	RTE (2-class)	CB (3-class)	TREC (6-class)
GPT-3 4-shot (from Zhao et al. (2021))	*	58.7 (11.9)	45.2 (19.4)	60.2 (7.6)
Calibrate Before Use (CBU) (Zhao et al., 2021)	*	60.4 (8.1)	60.7 (6.7)	69.7 (1.4)
Prompt-based FT (Gao et al., 2021)	*	52.8 (0.9)	84.4 (3.2)	54.8 (2.9)

Prompt-based FT [Gao et al. '21]: full DeBERTa fine-tuning with prompted inputs (uses 2 examples per class ⇒ 6 examples for CB and 12 examples for TREC)



Results: Few-shot

Using 4 labeled examples only

Model	View	RTE (2-class)	CB (3-class)	TREC (6-class)
GPT-3 4-shot (from Zhao et al. (2021))	*	58.7 (11.9)	45.2 (19.4)	60.2 (7.6)
Calibrate Before Use (CBU) (Zhao et al., 2021)	*	60.4 (8.1)	60.7 (6.7)	69.7 (1.4)
Prompt-based FT (Gao et al., 2021)	*	52.8 (0.9)	84.4 (3.2)	54.8 (2.9)
Label Model (no co-training)	ϕ_0	62.8	76.8	77.2
Label Model $\rightarrow$ DeBERTa distillation	ϕ_1	67.2 (0.5)	81.6 (2.2)	63.3 (0.4)

Results: Few-shot

Using 4 labeled examples only

Model	View	RTE (2-class)	CB (3-class)	TREC (6-class)
GPT-3 4-shot (from Zhao et al. (2021))	*	58.7 (11.9)	45.2 (19.4)	60.2 (7.6)
Calibrate Before Use (CBU) (Zhao et al., 2021)	*	60.4 (8.1)	60.7 (6.7)	69.7 (1.4)
Prompt-based FT (Gao et al., 2021)	*	52.8 (0.9)	84.4 (3.2)	54.8 (2.9)
Label Model (no co-training)	ϕ_0	62.8	76.8	77.2
Label Model $\rightarrow$ DeBERTa distillation	ϕ_1	67.2 (0.5)	81.6 (2.2)	63.3 (0.4)
Label Model + *co-training*	ϕ_0	64.9 (1.1)	83.5 (2.3)	78.3 (1.2)
DeBERTa-large + *co-training*	ϕ_1	**67.4** (2.3)	**86.2** (3.2)	**80.6** (1.1)

Results: Few-shot

Using 4 labeled examples only

Model	View	RTE (2-class)	CB (3-class)	TREC (6-class)
GPT-3 4-shot (from Zhao et al. (2021))	*	58.7 (11.9)	45.2 (19.4)	60.2 (7.6)
Calibrate Before Use (CBU) (Zhao et al., 2021)	*	60.4 (8.1)	60.7 (6.7)	69.7 (1.4)
Prompt-based FT (Gao et al., 2021)	*	52.8 (0.9)	84.4 (3.2)	54.8 (2.9)
Label Model (no co-training)	ϕ_0	62.8	76.8	77.2
Label Model $\rightarrow$ DeBERTa distillation	ϕ_1	67.2 (0.5)	81.6 (2.2)	63.3 (0.4)
Label Model + *co-training*	ϕ_0	64.9 (1.1)	83.5 (2.3)	78.3 (1.2)
DeBERTa-large + *co-training*	ϕ_1	**67.4** (2.3)	**86.2** (3.2)	**80.6** (1.1)

Same-sized models.

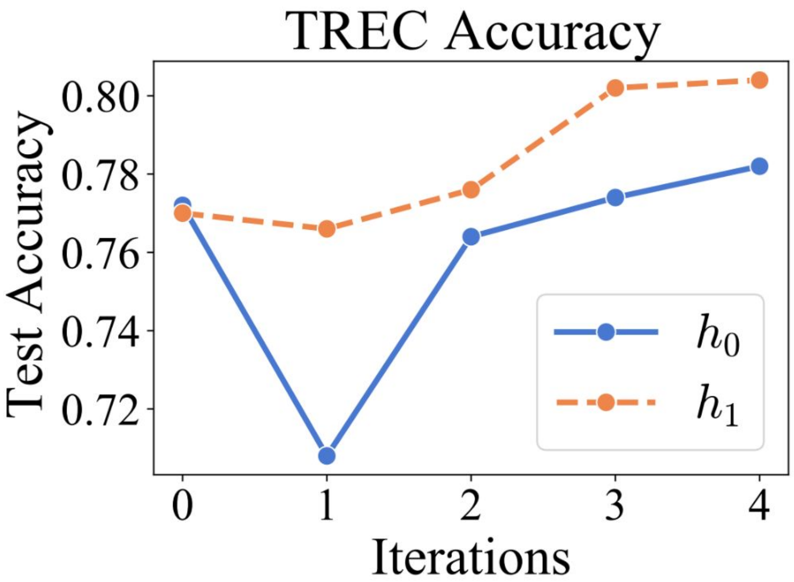
More than 100x smaller than GPT-3!

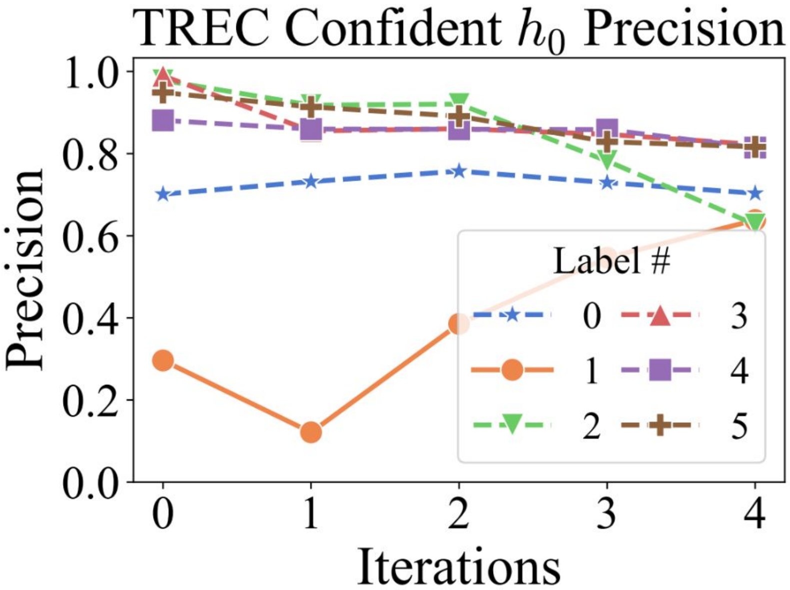
Results: Few-shot

Using 4 labeled examples only

Model	View	RTE (2-class)	CB (3-class)	TREC (6-class)
GPT-3 4-shot (from Zhao et al. (2021))	*	58.7 (11.9)	45.2 (19.4)	60.2 (7.6)
Calibrate Before Use (CBU) (Zhao et al., 2021)	*	60.4 (8.1)	60.7 (6.7)	69.7 (1.4)
Prompt-based FT (Gao et al., 2021)	*	52.8 (0.9)	84.4 (3.2)	54.8 (2.9)
Label Model (no co-training)	ϕ_0	62.8	76.8	77.2
Label Model $\rightarrow$ DeBERTa distillation	ϕ_1	67.2 (0.5)	81.6 (2.2)	63.3 (0.4)
Label Model + *co-training*	ϕ_0	64.9 (1.1)	83.5 (2.3)	78.3 (1.2)
DeBERTa-large + *co-training*	ϕ_1	**67.4** (2.3)	**86.2** (3.2)	**80.6** (1.1)
Label Model on full train	ϕ_0	67.8 (0.5)	82.7 (0.8)	91.9 (1.1)
DeBERTa-large on full train	ϕ_1	93.3	95.2	96.7
GPT-3 32-shot[†] (Brown et al., 2020)	*	69.0	75.6	*

Analysis


TREC Accuracy
Test Accuracy
0.80
0.78
0.76
0.74
0.72
0
1
2
3
4
Iterations
h_0
h_1


TREC Confident h_0 Precision
Precision
1.0
0.8
0.6
0.4
0.2
0.0
0
1
2
3
4
Iterations
Label #
0
1
2
3
4
5

Co-Training for Zero-shot Learning

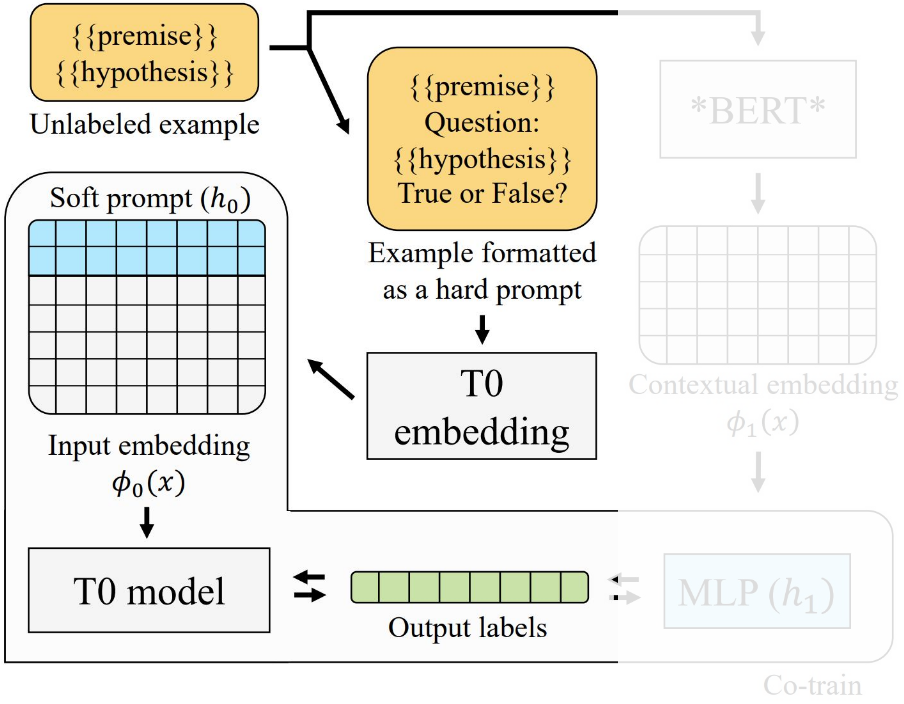
T0 [Sanh et al. ’21]: trained on tasks converted as natural instructions ⇒ meaningful zero-shot learning performance.

Co-Training for Zero-shot Learning

T0 [Sanh et al. '21]: trained on tasks converted as natural instructions ⇒ meaningful zero-shot learning performance.

$$h_0(\phi_0(X))$$

Soft prompt vectors appended to T0 word embeddings.

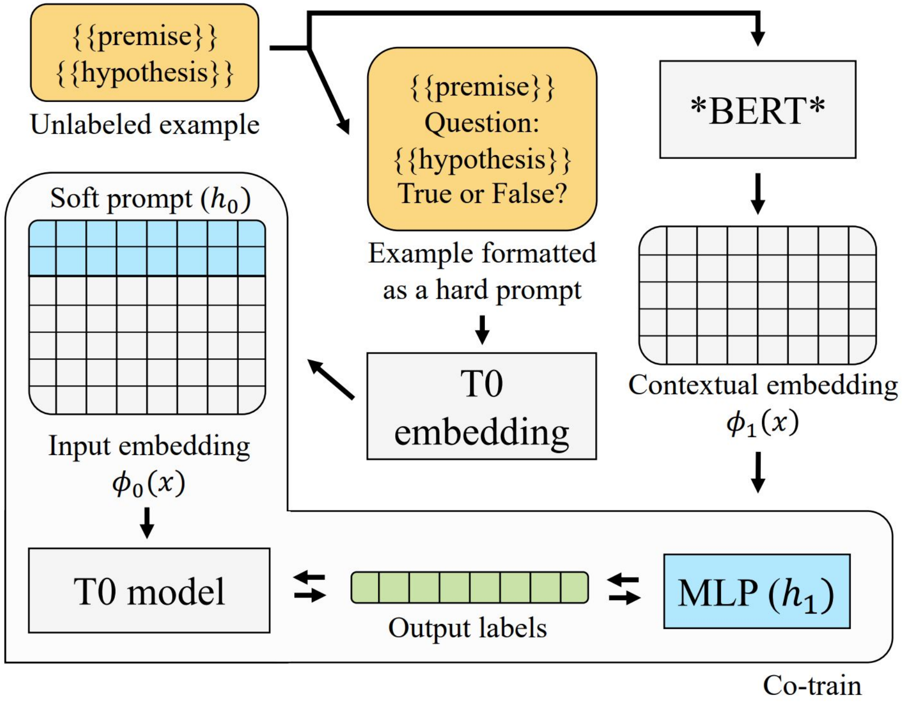


Co-Training for Zero-shot Learning

T0 [Sanh et al. '21]: trained on tasks converted as natural instructions ⇒ meaningful zero-shot learning performance.

$h_0(\phi_0(X))$ Soft prompt vectors appended to T0 word embeddings.

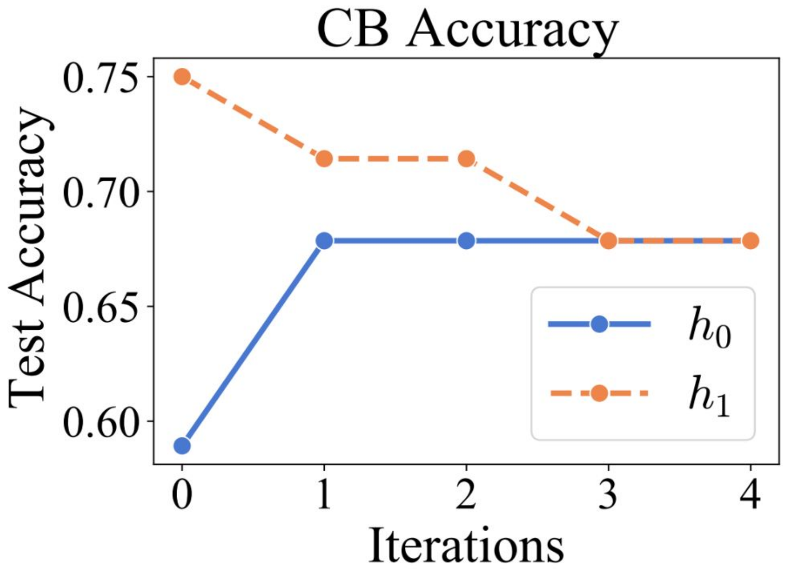
$h_1(\phi_1(X))$ DeBERTa + MLP classifier (same as before).



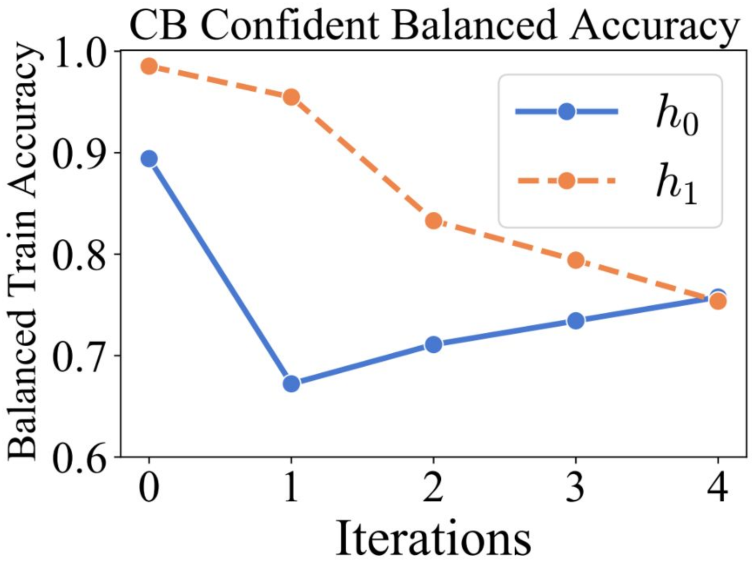
Results: Zero-shot

Model/Algorithm	View	RTE	CB	BoolQ
T0-3B (best) (Sanh et al., 2022)	ϕ_0	68.9	66.1	59.1
T0-3B zero-shot (no co-training)	ϕ_0	68.9	58.9	56.4
T0-3B soft prompt + *co-training*	ϕ_0	87.0	67.9	49.1
DeBERTa-large + *co-training*	ϕ_1	86.3	67.9	48.9
T0-3B soft prompt on full train	ϕ_0	90.6	80.4	86.9
DeBERTa-large on full train	ϕ_1	93.3	95.2	86.1

Analysis



CB Accuracy
Test Accuracy
0.75
0.70
0.65
0.60
0
1
2
3
4
Iterations
h_0
h_1



CB Confident Balanced Accuracy
Balanced Train Accuracy
1.0
0.9
0.8
0.7
0.6
0
1
2
3
4
Iterations
h_0
h_1

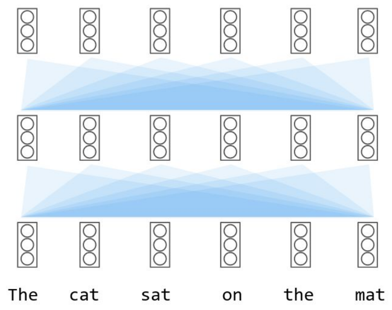
Summary

- Co-training can effectively distill few-shot and zero-shot capabilities from larger language models to much more efficient models.

- Future directions:

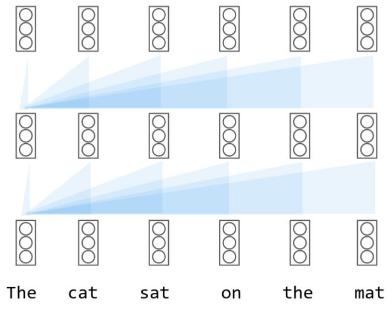
  - Extension to structured cases.
  - Co-training aware prompting.
  - Prompt-aware pretraining.

Efficient Transfer Learning with Language Models



Various notions of efficiency:

- Memory efficiency: parameters, storage cost
- Inference efficiency: FLOPs, energy, speed
- Data efficiency: labeled data, unlabeled data



Important to think about target use case when striving for efficiency!

Thanks!