석사학위논문심사 발표

Bayesian Transfer Learning for Soft Prompt Tuning

한국과학기술원 김재철AI대학원 석사과정

이 해 주

2023-06-13

Contents

Prompting in GPT-3

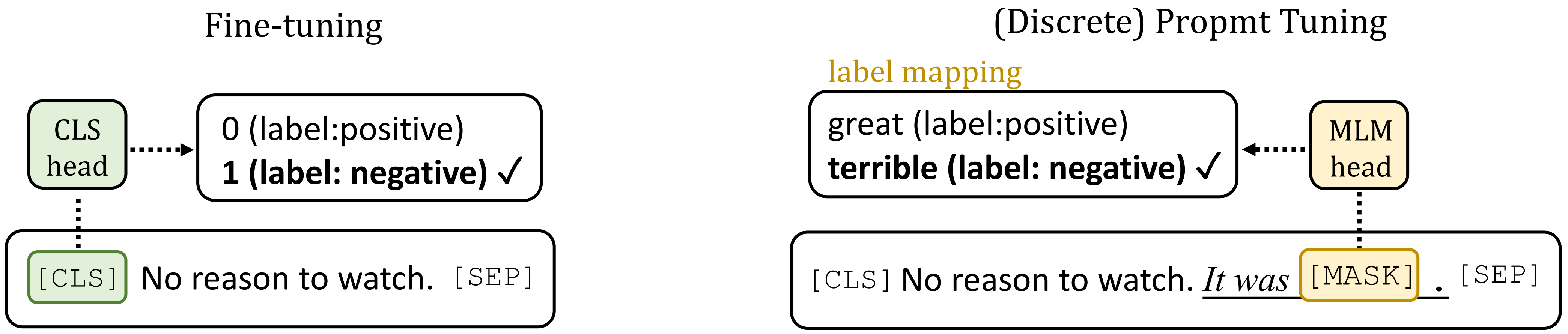
Input to GPT-3

```
Translate English to French:
Sea otter  =>  loutre de mer
Peppermint  =>  menthe poivrée
Plush giraffe  =>  giraffe peluche
Cheese  =>
```

prompts with demonstration examples

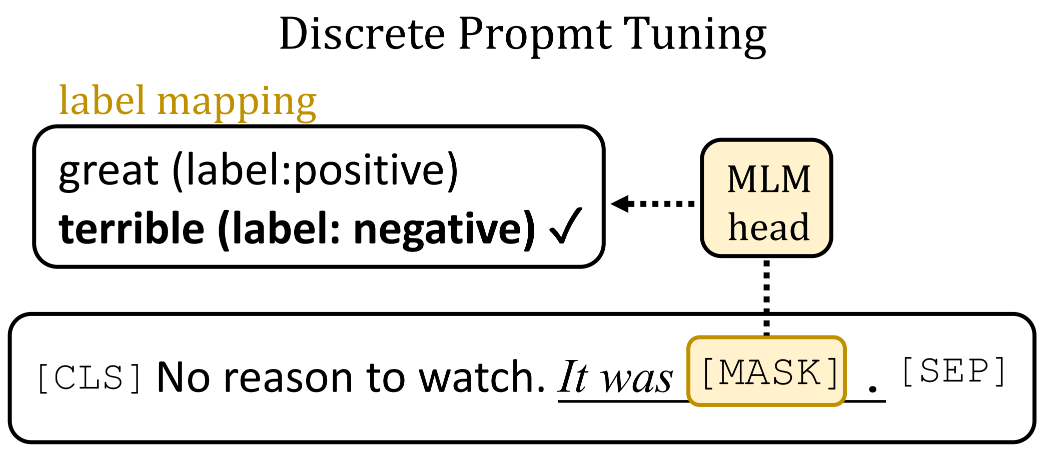
- Discovery of Prompting in GPT-3
  - We can use prompts to let Language Model (LM) solve arbitrary Natural Language Processing (NLP) tasks
  - Inspired by this discovery, many works exploring capabilities of “Prompting”, even in smaller models (GPT-3: 175B params), have been done

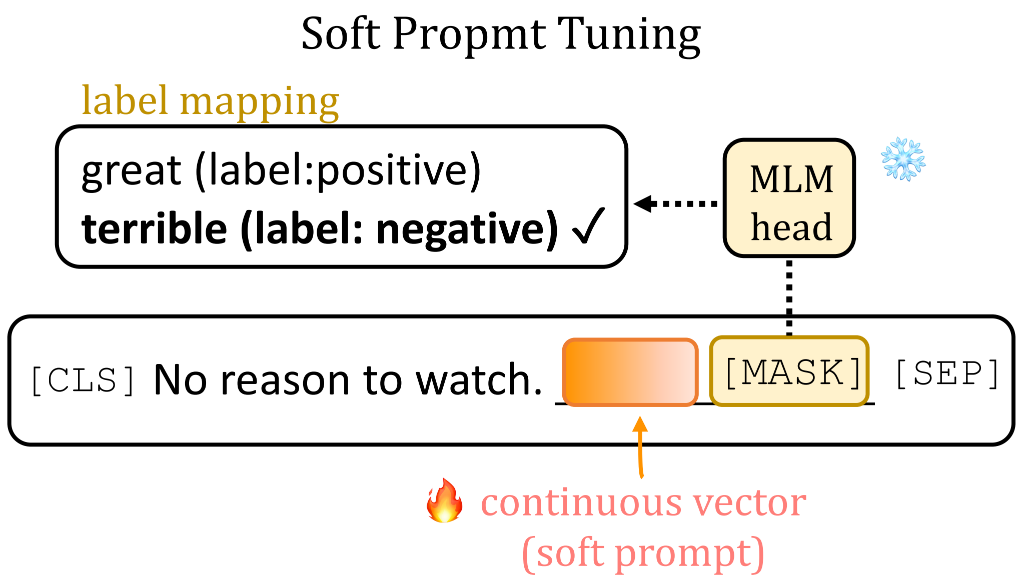
Prompt Tuning in NLP



- Fine-tuning vs. Prompt Tuning
  - Fine-tuning trains the whole LM to solve down-stream task
  - Prompt tuning adds “prompt” to input sequence, naturally invoking LM to solve down-stream task

Prompts as Continuous Vectors


Discrete Propmt Tuning
label mapping
great (label:positive)
terrible (label: negative) ✓
MLM head
[CLS] No reason to watch. It was [MASK] . [SEP]


Soft Propmt Tuning
label mapping
great (label:positive)
terrible (label: negative) ✓
MLM head
[CLS] No reason to watch. [MASK] [SEP]
continuous vector (soft prompt)

- Soft Prompt Tuning[1]
  - Prompt made up of discrete words may not be the optimal prompt
  - What if we replace the discrete words with sequence of continuous vectors, then optimize these prompts with gradient descent?
  - We call this approach "Soft Prompt Tuning"

[1] Lester et al., "The Power of Scale for Parameter-Efficient Prompt Tuning", EMNLP 2021

Pros and Cons of Soft Prompt Tuning

- Pros:
  - Parameter-efficient: Soft Prompt Tuning only updates soft prompt, while leaving LM untouched (frozen)
  - Space-efficient: a soft prompt can embed an NLP task; need small space for serving multiple NLP tasks
  - Easy to implement: compared to other Parameter-efficient methods[1,2,3], you only need to add soft prompt to the input
- Cons:
  - Long training time: compared to Fine-tuning, Soft Prompt Tuning suffers long and unstable training
  - Low performance: Soft Prompt Tuning underperforms compared to Fine-tuning

[1] Houlsby et al., "Parameter-Efficient Transfer Learning for NLP", ICML 2019
[2] Ben-Zaken et al., "BitFit: Simple Parameter-efficient Fine-tuning for Transformer-based Masked Language-models", ACL 2022
[3] Hu et al., "LoRA: Low-Rank Adaptation of Large Language Models", ICLR 2022

Transfer Learning in Soft Prompt Tuning

- To cope with the cons of Soft Prompt Tuning, several works[1,2,3] transfer soft prompt learned on source tasks to target task(s)
  - Transfer learning in Soft Prompt Tuning assumes transferability between NLP tasks, meaning that some NLP tasks help solving other similar NLP tasks
  - SPoT[1], ATTEMPT[2], and MPT[3] pre-train soft prompt on various source tasks, then fine-tune the soft prompt on target task(s)
    - They pre-train a soft prompt for each source task, and aggregate or retrieve the source task prompts to initialize target task prompt

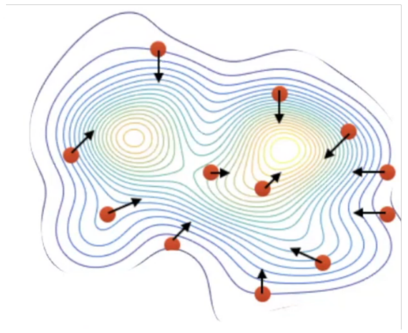
[1] Vu et al., "Better Frozen Model Adaptation through Soft Prompt Transfer", ACL 2022
[2] Asai et al., "ATTEMPT: Parameter-Efficient Multi-task Tuning via Attentional Mixtures of Soft Prompts", EMNLP 2022
[3] Wang et al., ”Multitask Prompt Tuning Enables Parameter-Efficient Transfer Learning", ICLR 2023

Stein Variational Gradient Descent (SVGD)

- SVGD[1] is a nonparametric variaional inference method
  - It converges fast to desired distribution due to its deterministic update rule, while not confined to tractable parametric distribution
  - Specifically, to obtain M samples from target distribution $p(\theta)$ where θ is neural net parameters, SVGD employs M instances of θ, $\Theta = \{\theta^m\}_{m=1}^M$, called *particles*
  - At iteration t, each particle $\theta_t \in \Theta_t$ is updated by following rule:

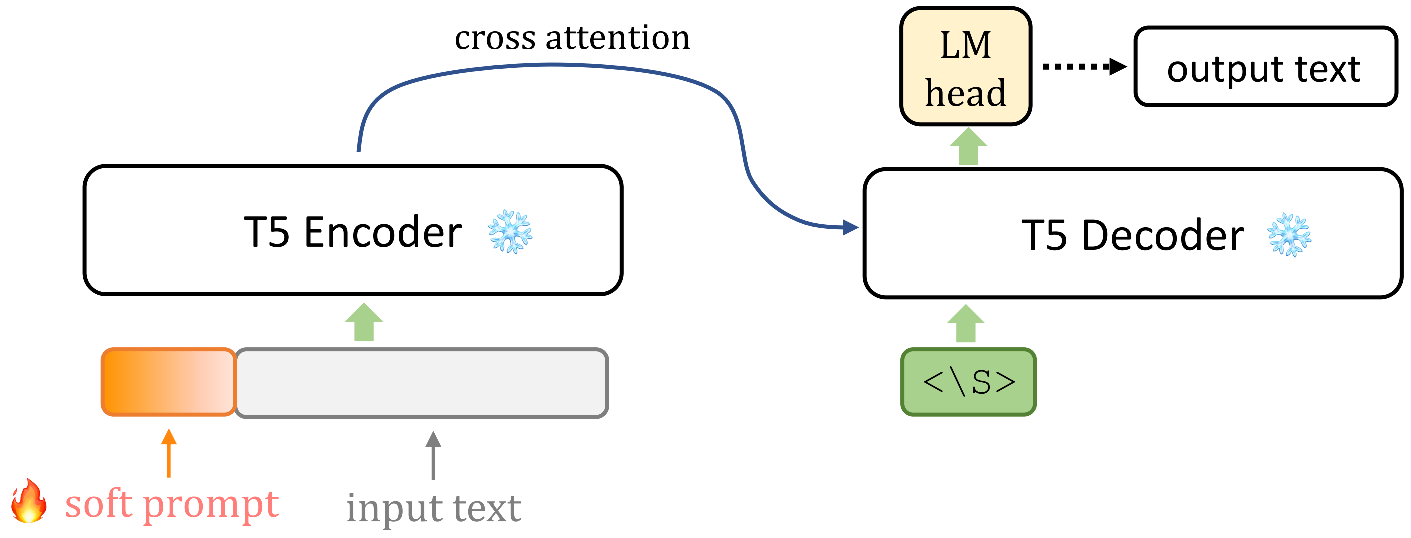
$$\theta_{t+1} \leftarrow \theta_t + \epsilon_t \phi(\theta_t) \quad \text{where} \quad \phi(\theta_t) = \frac{1}{M} \sum_{j=1}^{M} \left[k(\theta_t^j, \theta_t) \nabla_{\theta_t^j} \log p(\theta_t^j) + \nabla_{\theta_t^j} k(\theta_t^j, \theta_t) \right]$$

where ϵ_t is step-size and $k(x, x')$ is a positive-definite kernel (e.g. RBF)



[1] Liu et al., "Stein Variational Gradient Descent: A General Purpose Bayesian Inference Algorithm", NeurIPS 2016

Problem Setting & Notation



- Like previous works[1,2,3], we use T5 model for Soft Prompt Tuning
  - Prepend soft prompt to input text
  - Experiments conducted with T5-base, T5-large, and T5-XL
- Train and evaluate on various NLP tasks, including GLUE and SuperGLUE

[1] Vu et al., "Better Frozen Model Adaptation through Soft Prompt Transfer", ACL 2022
[2] Asai et al., "ATTEMPT: Parameter-Efficient Multi-task Tuning via Attentional Mixtures of Soft Prompts", EMNLP 2022
[3] Wang et al., "Multitask Prompt Tuning Enables Parameter-Efficient Transfer Learning", ICLR 2023

Problem Setting & Notation (cond.)

- Notations
  - $\boldsymbol{\theta}^{\mathcal{S}}$ denotes source task prompt, $\boldsymbol{\theta}^{\mathcal{T}}$ denotes target task prompt
  - $\mathcal{D}_k^{\mathcal{S}}$ and $\mathcal{D}_k^{\mathcal{T}}$ denote the k-th task data of source tasks, k-th task data of target tasks (i.e. i–th datapoint of $\mathcal{D}_k^{\mathcal{S}}$ is $(\mathbf{Y}_i^k, \mathbf{X}_i^k)$, where X and Y are input text and label)
  - If we want to optimize a prompt w.r.t. k-th source task, we usually perform Maximum Likelihood Estimation (MLE)

$$\underset{\boldsymbol{\theta}}{\operatorname{argmax}} \prod_{i=1}^{|\mathcal{D}_k^{\mathcal{S}}|} \mathbb{P}_{\mathrm{LM}}(\mathbf{Y}_i^k \mid [\boldsymbol{\theta}; \mathbf{X}_i^k])$$

  - For simplify notation, we use $\mathbb{P}(\mathcal{D}_k^{\mathcal{S}}|\boldsymbol{\theta})$ instead of $\prod_i \mathbb{P}_{\mathrm{LM}}(\mathbf{Y}_i^k|[\boldsymbol{\theta}; \mathbf{X}_i^k])$
  - When the prior of each source task is uninformative (uniform), MLE coincides with MAP, as $\mathbb{P}(\boldsymbol{\theta}|\mathcal{D}_k^{\mathcal{S}}) \propto \mathbb{P}(\mathcal{D}_k^{\mathcal{S}}|\boldsymbol{\theta})\,\mathbb{P}(\boldsymbol{\theta})$

Problem Setting & Notation (cond.)

- Notations (cond.)
  - Under transfer learning paradigm, our aim at the source task prompt training stage is to maximize the *global posterior distribution* over source tasks, which is the product of individual source task posteriors (under uniform priors):

$$\mathbb{P}(\boldsymbol{\theta} \,|\, \mathcal{D}^{\mathcal{S}}) = \prod_{k=1}^{K} \mathbb{P}(\boldsymbol{\theta} \,|\, \mathcal{D}_k^{\mathcal{S}})$$

  - After source prompt training stage, $\boldsymbol{\theta}^{\mathcal{S}}$ is obtained, and it is used to initialize $\boldsymbol{\theta}^{\mathcal{T}}$. Note that $\boldsymbol{\theta}^{\mathcal{S}}$ is derived from $\mathcal{D}^{\mathcal{S}}$.
  - Therefore, we can express the target task prompt adaptation as follows:

$$\underset{\boldsymbol{\theta}^{\mathcal{T}}}{\operatorname{argmax}}\, \mathbb{P}(\mathcal{D}^{\mathcal{T}} \,|\, \boldsymbol{\theta}^{\mathcal{T}})\, \mathbb{P}(\boldsymbol{\theta}^{\mathcal{T}} \,|\, \mathcal{D}^{\mathcal{S}}) \quad \cdots \text{ Eq. 1}$$

  this can be considered as product of likelihood and prior over $\boldsymbol{\theta}^{\mathcal{T}}$

Issue with Previous Works

- Previous works either 1) retrive the most relevant source prompt to target task[1], 2) takes attentional mixture of source prompts[2], or 3) average source prompts[3], to initialize target task prompt.
- We can easily show that previous works fail, even with cases where the posteriors of individual source tasks are simple Gaussian
  - Suppose $\mathbb{P}(\boldsymbol{\theta}|\mathcal{D}_k^{\mathcal{S}})$ has mean of $\boldsymbol{\mu}_k$ and covariance of $\boldsymbol{\Sigma}_k$
  - If we perform MLE (since we assume uniform prior), the prompt of k-th source task will be close to $\boldsymbol{\mu}_k$
  - A product of multivariate Gaussians is proportional to a Gaussian, and the global posterior distribution has mean of :

$$\boldsymbol{\mu} = \left(\sum_{k=1}^{K} \boldsymbol{\Sigma}_k^{-1}\right)^{-1}\left(\sum_{k=1}^{K} \boldsymbol{\Sigma}_k^{-1}\boldsymbol{\mu}_k\right)$$

[1] Vu et al., "Better Frozen Model Adaptation through Soft Prompt Transfer", ACL 2022
[2] Asai et al., "ATTEMPT: Parameter-Efficient Multi-task Tuning via Attentional Mixtures of Soft Prompts", EMNLP 2022
[3] Wang et al., ”Multitask Prompt Tuning Enables Parameter-Efficient Transfer Learning", ICLR 2023

Issue with Previous Works (cond.)

- We can easily show that previous works fail, even with cases where the posteriors of individual source tasks are simple Gaussian (cond.)
  - At transfer learning, we hope that sufficient data from the source tasks will lead to a good initialization point for the target task
  - However, unless covariances are identity matrices, it is unlikely that any convex combination of $\boldsymbol{\mu}_k$ will equal $\boldsymbol{\mu}$
  - Therefore, we cannot expect the weighted-average of source prompts to perform well on arbitrary target task

Formulation of Prompt Optimization Process

- We approximate global posterior of source tasks via SVGD, then leverage the approximate distribution to initialize target task prompt
  - The target task adaptation can be modeled as Maximum A Posteriori (MAP), as shown in Eq. 1
  - In Eq. 1, theta $\boldsymbol{\theta}^{\mathcal{T}}$ is initialized from $\boldsymbol{\theta}^{\mathcal{S}}$, and $\boldsymbol{\theta}^{\mathcal{S}}$ is derived from $\mathcal{D}^{\mathcal{S}}$; therefore we can re-formulate Eq. 1 without argmax as follows:

$$\mathbb{P}(\mathcal{D}^{\mathcal{T}}|\boldsymbol{\theta}^{\mathcal{T}}) \int_{\boldsymbol{\theta}^{\mathcal{S}}} \mathbb{P}(\boldsymbol{\theta}^{\mathcal{T}}|\boldsymbol{\theta}^{\mathcal{S}})\, \mathbb{P}(\boldsymbol{\theta}^{\mathcal{S}}|\mathcal{D}^{\mathcal{S}}) d\boldsymbol{\theta}^{\mathcal{S}}$$

Eq. 1

$$\underset{\boldsymbol{\theta}^{\mathcal{T}}}{\operatorname{argmax}}\, \mathbb{P}(\mathcal{D}^{\mathcal{T}}\,|\,\boldsymbol{\theta}^{\mathcal{T}})\, \mathbb{P}(\boldsymbol{\theta}^{\mathcal{T}}\,|\,\mathcal{D}^{\mathcal{S}})$$

  - The above term can also be written as:

$$\mathbb{P}(\mathcal{D}^{\mathcal{T}}|\boldsymbol{\theta}^{\mathcal{T}}) \cdot \underset{\boldsymbol{\theta}^{\mathcal{S}}\sim\mathbb{P}(\cdot|\mathcal{D}^{\mathcal{S}})}{\mathbb{E}} \left[\mathbb{P}(\boldsymbol{\theta}^{\mathcal{T}}|\boldsymbol{\theta}^{\mathcal{S}})\right] \overset{\substack{\text{MC} \\ \text{Sampling}}}{\simeq} \mathbb{P}(\mathcal{D}^{\mathcal{T}}|\boldsymbol{\theta}^{\mathcal{T}}) \cdot \frac{1}{M}\sum_{i=1}^{M} \mathbb{P}(\boldsymbol{\theta}^{\mathcal{T}}|\boldsymbol{\theta}_i^{\mathcal{S}})$$

Formulation of Prompt Optimization Process (cond.)

- We approximate global posterior of source tasks via SVGD, then leverage the approximate distribution to initialize target task prompt (cond.)
  - How can we sample $\boldsymbol{\theta}^{\mathcal{S}}$ from $\mathbb{P}(\,\cdot\,|\mathcal{D}^{\mathcal{S}})$? → Use SVGD!
  - The SVGD update during source task training would be:

$$\begin{aligned}\phi(\theta) &= \frac{1}{M}\sum_{j=1}^{M}\left[k(\theta_j,\theta)\nabla_{\theta_j}\log\mathbb{P}(\theta_j|\mathcal{D}^{\mathcal{S}}) + \nabla_{\theta_j}k(\theta_j,\theta)\right] \\ &= \frac{1}{M}\sum_{j=1}^{M}\left[k(\theta_j,\theta)\nabla_{\theta_j}\log\mathbb{P}(\mathcal{D}^{\mathcal{S}}|\theta_j) + \nabla_{\theta_j}k(\theta_j,\theta)\right]\end{aligned}$$

$$\left(\because \nabla_{\theta_j}\log\{\mathbb{P}(\mathcal{D}^{\mathcal{S}}|\theta_j)\cdot \overset{\text{uniform prior}}{\overset{\downarrow}{\mathbb{P}(\theta_j)}}\} = \nabla_{\theta_j}\log\mathbb{P}(\mathcal{D}^{\mathcal{S}}|\theta_j)\right)$$

  - Therefore, we empirically use the minus of the gradient of LM loss for $\nabla_{\theta_j}\log\mathbb{P}(\mathcal{D}^{\mathcal{S}}|\theta_j)$

Note: SVGD update rule

$$\theta_{t+1} \leftarrow \theta_t + \epsilon_t\phi(\theta_t) \quad \text{where} \quad \phi(\theta_t) = \frac{1}{M}\sum_{j=1}^{M}\left[k(\theta_t^j,\theta_t)\nabla_{\theta_t^j}\log p(\theta_t^j) + \nabla_{\theta_t^j}k(\theta_t^j,\theta_t)\right]$$

Formulation of Prompt Optimization Process (cond.)

- Taking $\max$ operation to Eq. 1 derives the following:
  - $$\max_{\boldsymbol{\theta}^{\mathcal{T}}} \log(\mathbb{P}(\mathcal{D}^{\mathcal{T}} \mid \boldsymbol{\theta}^{\mathcal{T}})\,\mathbb{P}(\boldsymbol{\theta}^{\mathcal{T}} \mid \mathcal{D}^{\mathcal{S}})) \simeq \min_{\boldsymbol{\theta}^{\mathcal{T}}} -\log \mathbb{P}(\mathcal{D}^{\mathcal{T}} | \boldsymbol{\theta}^{\mathcal{T}}) - \log \frac{1}{M} \sum_{i=1}^{M} \mathbb{P}(\boldsymbol{\theta}^{\mathcal{T}} | \boldsymbol{\theta}_i^{\mathcal{S}})$$
    $$\leq \min_{\boldsymbol{\theta}^{\mathcal{T}}} -\log \mathbb{P}(\mathcal{D}^{\mathcal{T}} | \boldsymbol{\theta}^{\mathcal{T}}) - \frac{1}{M} \sum_{i=1}^{M} \log \mathbb{P}(\boldsymbol{\theta}^{\mathcal{T}} | \boldsymbol{\theta}_i^{\mathcal{S}}) \quad \cdots \text{Eq. 2}$$
  - We minimize the *upper bound* of Eq. 2 to maximize Eq. 1
  - We design $\mathbb{P}(\boldsymbol{\theta}^{\mathcal{T}} | \boldsymbol{\theta}_i^{\mathcal{S}})$ in the second term of upper bound, with simple Gaussian distribution with mean of $\boldsymbol{\theta}_i^{\mathcal{S}}$ and constant variance of $\Sigma_i = \sigma^2 \mathbf{I}$, since we lack prior information about the target task before adaptation

Eq. 1

$$\operatorname*{argmax}_{\boldsymbol{\theta}^{\mathcal{T}}} \mathbb{P}(\mathcal{D}^{\mathcal{T}} \mid \boldsymbol{\theta}^{\mathcal{T}})\,\mathbb{P}(\boldsymbol{\theta}^{\mathcal{T}} \mid \mathcal{D}^{\mathcal{S}})$$

Formulation of Prompt Optimization Process (cond.)

- Taking $\max$ operation to Eq. 1 derives the following (cond.):
  - $$\max_{\boldsymbol{\theta}^{\mathcal{T}}} \mathbb{P}(\mathcal{D}^{\mathcal{T}} \,|\, \boldsymbol{\theta}^{\mathcal{T}})\, \mathbb{P}(\boldsymbol{\theta}^{\mathcal{T}} \,|\, \mathcal{D}^{\mathcal{S}}) \simeq \min_{\boldsymbol{\theta}^{\mathcal{T}}} - \log \mathbb{P}(\mathcal{D}^{\mathcal{T}} | \boldsymbol{\theta}^{\mathcal{T}}) - \log \frac{1}{M} \sum_{i=1}^{M} \mathbb{P}(\boldsymbol{\theta}^{\mathcal{T}} | \boldsymbol{\theta}_i^{\mathcal{S}})$$
    $$\leq \min_{\boldsymbol{\theta}^{\mathcal{T}}} - \log \mathbb{P}(\mathcal{D}^{\mathcal{T}} | \boldsymbol{\theta}^{\mathcal{T}}) - \frac{1}{M} \sum_{i=1}^{M} \log \mathbb{P}(\boldsymbol{\theta}^{\mathcal{T}} | \boldsymbol{\theta}_i^{\mathcal{S}}) \quad \cdots \text{Eq. 2}$$
  - We minimize the *upper bound* of Eq. 2 to maximize Eq. 1 (cond.)
  - By introducing additive constant C, the second term of upper bound can be rewritten as:
    $$\frac{1}{2\sigma^2} \left\| \boldsymbol{\theta}^{\mathcal{T}} - \frac{1}{M} \sum_{i=1}^{M} \boldsymbol{\theta}_i^{\mathcal{S}} \right\|^2 + C$$
  - Therefore, we initialize $\boldsymbol{\theta}^{\mathcal{T}}$ with $\frac{1}{M} \sum_{i=1}^{M} \boldsymbol{\theta}_i^{\mathcal{S}}$, to ensure we start target adaptation from the point of maximum prior probability

Target Adaptation

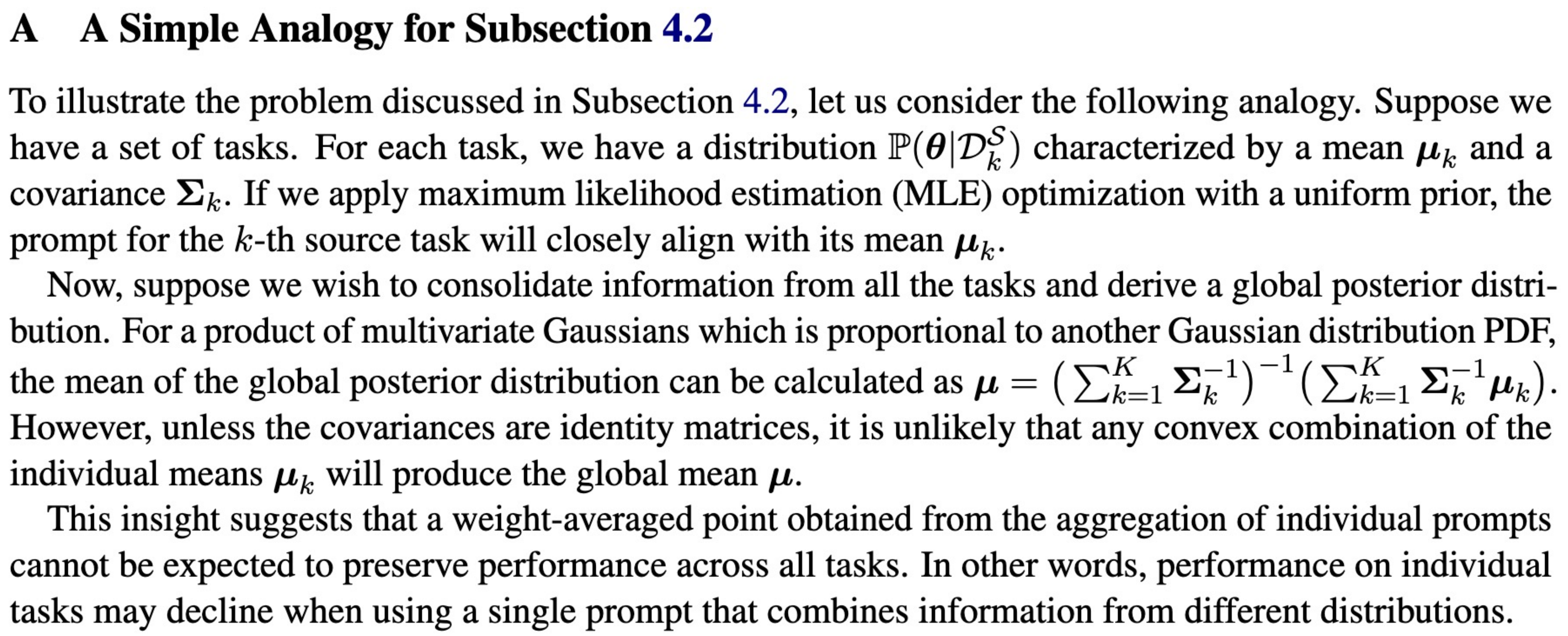
- After initializing target prompt as $\boldsymbol{\theta}^{\mathcal{T}} \leftarrow \frac{1}{M} \sum_{i=1}^{M} \boldsymbol{\theta}_i^{\mathcal{S}}$, we perform simple SGD on target task dataset
  - SGD with respect to LM loss & regularization (log of gaussian, as form of weight decay)

Experiment Result (preliminary)

Glue	MNLI	QQP	QNLI	SST-2	STS-B	MRPC	RTE	CoLA	Avg.
MPT	85.9	90.3	93.1	93.8	90.4	89.1	79.4	62.4	85.6
Ours	83.9	93.5	93.1	94.3	89.7	90.0	81.5	79.8	88.2

Superglue	BoolQ	Multirc(f1)	WiC	WSC	CB	Avg.
MPT	79.6	74.8	69	67.3	79.8	74.1
Ours	78.4	71.2	69.4	67.3	85.7	74.4

Appendix A.



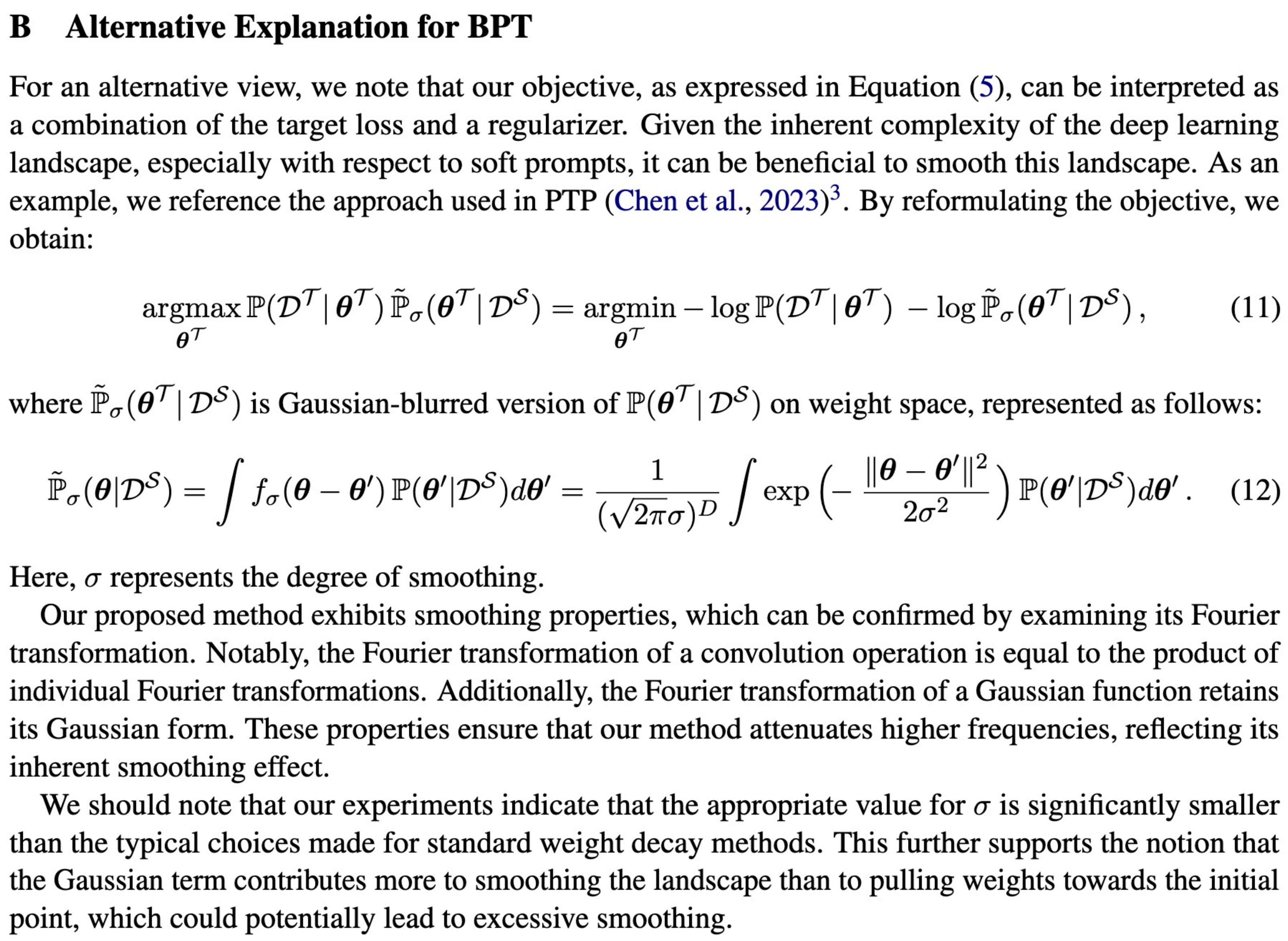
A A Simple Analogy for Subsection 4.2

To illustrate the problem discussed in Subsection 4.2, let us consider the following analogy. Suppose we have a set of tasks. For each task, we have a distribution $\mathbb{P}(\boldsymbol{\theta}|\mathcal{D}_k^{\mathcal{S}})$ characterized by a mean $\boldsymbol{\mu}_k$ and a covariance $\boldsymbol{\Sigma}_k$. If we apply maximum likelihood estimation (MLE) optimization with a uniform prior, the prompt for the k-th source task will closely align with its mean $\boldsymbol{\mu}_k$.

Now, suppose we wish to consolidate information from all the tasks and derive a global posterior distribution. For a product of multivariate Gaussians which is proportional to another Gaussian distribution PDF, the mean of the global posterior distribution can be calculated as $\boldsymbol{\mu} = \left(\sum_{k=1}^{K} \boldsymbol{\Sigma}_k^{-1}\right)^{-1}\left(\sum_{k=1}^{K} \boldsymbol{\Sigma}_k^{-1}\boldsymbol{\mu}_k\right)$. However, unless the covariances are identity matrices, it is unlikely that any convex combination of the individual means $\boldsymbol{\mu}_k$ will produce the global mean $\boldsymbol{\mu}$.

This insight suggests that a weight-averaged point obtained from the aggregation of individual prompts cannot be expected to preserve performance across all tasks. In other words, performance on individual tasks may decline when using a single prompt that combines information from different distributions.

Appendix B.



B Alternative Explanation for BPT

For an alternative view, we note that our objective, as expressed in Equation (5), can be interpreted as a combination of the target loss and a regularizer. Given the inherent complexity of the deep learning landscape, especially with respect to soft prompts, it can be beneficial to smooth this landscape. As an example, we reference the approach used in PTP (Chen et al., 2023)[3]. By reformulating the objective, we obtain:

$$\underset{\boldsymbol{\theta}^{\mathcal{T}}}{\operatorname{argmax}}\, \mathbb{P}(\mathcal{D}^{\mathcal{T}} \,|\, \boldsymbol{\theta}^{\mathcal{T}})\, \tilde{\mathbb{P}}_{\sigma}(\boldsymbol{\theta}^{\mathcal{T}} \,|\, \mathcal{D}^{\mathcal{S}}) = \underset{\boldsymbol{\theta}^{\mathcal{T}}}{\operatorname{argmin}} - \log \mathbb{P}(\mathcal{D}^{\mathcal{T}} \,|\, \boldsymbol{\theta}^{\mathcal{T}}) \; - \log \tilde{\mathbb{P}}_{\sigma}(\boldsymbol{\theta}^{\mathcal{T}} \,|\, \mathcal{D}^{\mathcal{S}})\,, \qquad (11)$$

where $\tilde{\mathbb{P}}_{\sigma}(\boldsymbol{\theta}^{\mathcal{T}} \,|\, \mathcal{D}^{\mathcal{S}})$ is Gaussian-blurred version of $\mathbb{P}(\boldsymbol{\theta}^{\mathcal{T}} \,|\, \mathcal{D}^{\mathcal{S}})$ on weight space, represented as follows:

$$\tilde{\mathbb{P}}_{\sigma}(\boldsymbol{\theta}|\mathcal{D}^{\mathcal{S}}) = \int f_{\sigma}(\boldsymbol{\theta} - \boldsymbol{\theta}')\, \mathbb{P}(\boldsymbol{\theta}'|\mathcal{D}^{\mathcal{S}}) d\boldsymbol{\theta}' = \frac{1}{(\sqrt{2\pi}\sigma)^{D}} \int \exp\left(-\frac{\|\boldsymbol{\theta} - \boldsymbol{\theta}'\|^2}{2\sigma^2}\right) \mathbb{P}(\boldsymbol{\theta}'|\mathcal{D}^{\mathcal{S}}) d\boldsymbol{\theta}'\,. \qquad (12)$$

Here, σ represents the degree of smoothing.

Our proposed method exhibits smoothing properties, which can be confirmed by examining its Fourier transformation. Notably, the Fourier transformation of a convolution operation is equal to the product of individual Fourier transformations. Additionally, the Fourier transformation of a Gaussian function retains its Gaussian form. These properties ensure that our method attenuates higher frequencies, reflecting its inherent smoothing effect.

We should note that our experiments indicate that the appropriate value for σ is significantly smaller than the typical choices made for standard weight decay methods. This further supports the notion that the Gaussian term contributes more to smoothing the landscape than to pulling weights towards the initial point, which could potentially lead to excessive smoothing.