GPT Understands, Too. paper review

검색

GPT가 학습하는 방법

2023. 05. 02 이세영 발표

CONTENTS

기본 (굵으면 강조)

엄청 중요

중요한 것, 소제목 등

부가설명 (포스트잇도 활용)

P-tuning이란?

- Fine-tuning 없이 Downstream task의 성능을 높이기 위한 pre-training 기법
- Continuous space 에서 프롬프트를 자동으로 검색하는 방법

프롬프트: 모델에게 질문이나 지시를 주는 문장

NN은 continuous 하므로 continuous 프롬프트를 사용해야 한다!

Fine-tuning 이란?

사전학습된 모델에 내가 하고자 하는 task에 맞춰 새로운 데이터로 추가로 학습시키는 방법

ex. BERT에 악성댓글 데이터 학습 -> 악성댓글 분류

Downstream task 란?

사전학습된 모델로 내가 최종적으로 하고자 하는 작업

ex. GLUE benchmark 의 task: CoLA, SST-2, MRPC, STS-B 등

NN은 Continuous 하다

NN은 연속함수이다.

$$L_1(L_2(\dots L_n(X)))$$

모든 구성요소가 연속적임.

Activation function, layer 등이 연속적임.

1. Introduction

P-tuning이 제안된 배경

- NLP에서 Pre-trained model은 성공적인 접근 방식
- Pre-training 동안 LM(Language Model)은 문법, 구문, 상식 등 학습
- LM 은 크게 3가지로 분류 가능
  - Unidirectional Language Model (ex. GPT)
  - Bidirectional Language Model (ex. BERT)
  - Hybrid Model (ex. XLNet)
- GPT스타일 모델이 fine-tuning이 있는 NLU작업에 대해 BERT보다 성능이 ↓

  Natural Language Understand

  => 따라서 **GPT가 NLU에 적합하지 않다**고 가정
- GPT-3를 활용한 few-shot, zero-shot 성능이 매우 좋았다.

  => 적절한 프롬프트를 사용한다면 GPT의 성능이 BERT보다 높지 않을까?

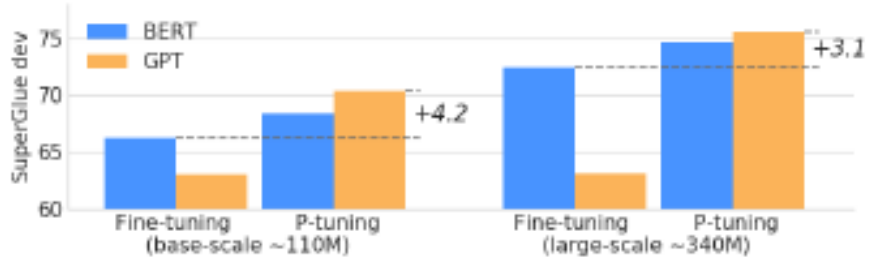


Figure 1. Average scores on 7 dev datasets of SuperGlue. GPTs can be better than similar-sized BERTs on NLU with P-tuning.

Few-shot 예시

Input
강아지: dog
고양이: cat
가족:

Output
강아지: dog
고양이: cat
가족: family

1. Introduction

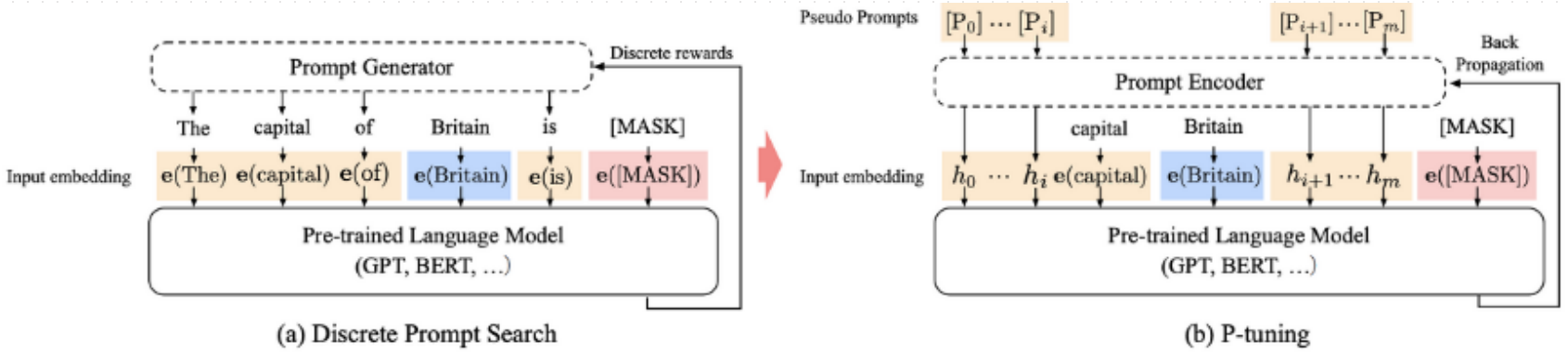
이전의 prompt 방식

- 이전 연구에서는 discrete prompt(이산 프롬프트) 를 사용하거나 적절한 프롬프트를 수동으로 탐색
- 최근에는 discrete prompt 자동 검색에 초점 맞춰 연구
- BUT! 신경망은 본질적으로 연속적이기 때문에, 적절한 continuous prompts 를 사용해야 함.

Discrete Prompt 와 Continuous Prompts 의 차이

Prompt Encoder 사용

	Discrete Prompt	Continuous Prompt
Template	The capital of Britain is [MASK]	
예시	$\{e(The)\ e(capital)$ $e(\text{of})\ e(Britain)\ e(is)\ e([MASK])\}$	$\{h_0 \cdots\ h_1\ e(capital)\ e(Britain)$ $h_{i+1} \cdots\ h_m\ e([MASK])\}$



(a) Discrete Prompt Search (b) P-tuning

2. P-tuning

Architecture

Notation

$\mathcal{M}$: $model$

$e(word)$: $word$ 의 embedding

$e \in \mathcal{M}$: e 는 $\mathcal{M}$의 embedding layer

$x_{1:n} = \{x_0, x_1, \ldots, x_n\}$: input tokens

p : $function\ of\ a\ prompt$

T: template

$\mathcal{V}$: $vocabulary\ of\ model\ \mathcal{M}$

$[P_i]$: i번째 prompt token in a template T

Scenario

수도 맞추기 문제

template: "The capital of Britain is [MASK]"

prompt: "The capital of … is … ."

Context: "Britain"

target: "[MASK]"

context와 target은 각 context 와 target 에 따라 변화가능

2. P-tuning

Architecture

Notation

$\mathcal{M}$: mode

$e(word)$: v

$e \in \mathcal{M} : e$

$x_{1:n} = \{x_0$

p : functi

T: templat

$\mathcal{V}$: vocabu

$[P_i]$: i번째

$\hat{h}_{0:m} = ar$

$h_i = MLP([LSTM(h_{0:i}); LSTM(h_{i:m})])$

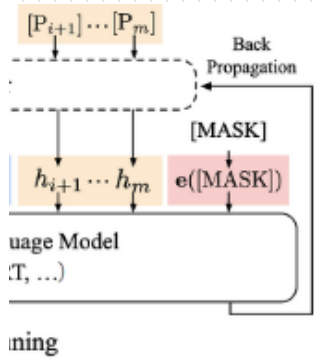
- Prompt Encoder 구성

```
PromptEncoder(
  (lstm): LSTM(hidden_size, hidden_size // 2, num_layers=2, bidirectional=True)
  (mlp): Sequential(
    (0): Linear(in_features=hidden_size, out_features=hidden_size)
    (1): ReLU()
    (2): Linear(in_features=hidden_size, out_features=hidden_size)
  )
)
```

- Bi-directional LSTM + MLP 구조

Scenario

수도 맞추기 문제



), $e(y)\}$

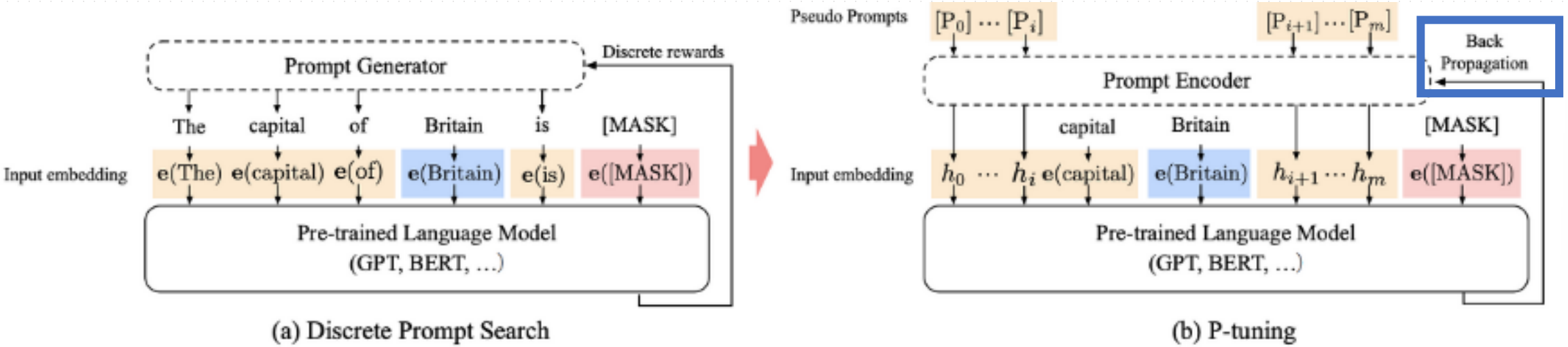
$e(is)\ e([MASK])\}$

., h_m, $e(y)\}$

→ *pesudo token*을 embed 하는 모델인 prompt encoder

bidirection LSTM 에 2-layer MLP 를 얹은 모델 사용 (MLP만 사용해도 됨)

2. P-tuning



Discrete prompts: $\{e([P_{0:i}]), \mathrm{e(x)}, \mathrm{e}([P_{i+1:m}]),\ e(y)\}$

P-tuning: $\{h_0, \ldots, h_i, e(capital), \mathrm{e(x)}, h_{i+1}, \ldots, h_m,\ e(y)\}$

anchor token

$$h_i = MLP([LSTM(h_{0:i}); LSTM(h_{i:m})])$$

$$\hat{h}_{0:m} = arg\min_h \mathcal{L}(\mathcal{M}(x, y))$$

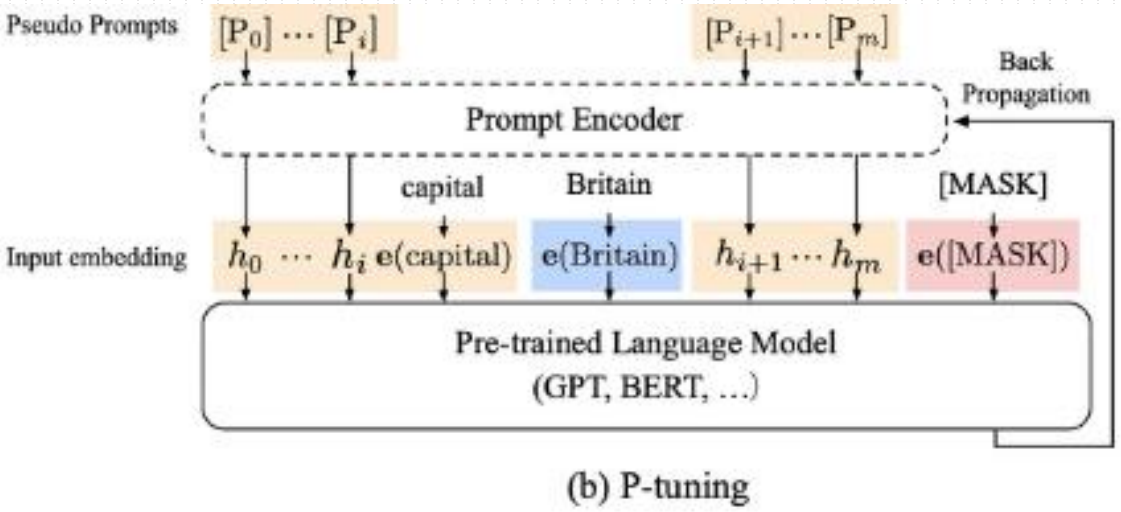
Pseudo prompts $[P_i]$ 를 h_i로 표현함.

inference 에서는 $\hat{h}_{0:m}$ 이 학습되어 고정됨.

2. P-tuning

Anchor Token

- Anchor Token: 모델에게 task 를 알려주는 token



(b) P-tuning

[Prediction a country's capital]

The capital of Britain is [MASK]

Anchor token: capital, 수도를 답해야 함을 알려줌

[RTE (Recognizing Textual Entailment) task]

[PRE][prompt tokens][HYP]?[prompt tokens][MASK]

Anchor token: ?, [HYP]가 질문으로 작용함을 나타냄

3. Experiments

LAMA dataset

Knowledge Probing Task

ex. Dante was born in [MASK]

output: Florence, Italy, korea, seoul, …

- Dataset
  - LAMA-34k: BERT의 모든 vocab 을 다루고 있음
  - LAMA-29k: GPT와 BERT의 공통 vocab을 다룸

- Template

숫자는 prompt token의 수

  - Bidirectional model(BERT): [3, sub, 3, obj, 3]
  - Unidirectional model(GPT): [3, sub, 3, obj]

- Setting
  - Anchor token X
  - Learning rate: 1e-5
  - Adam optimizer

- Tuning Method
  - Manual Prompt(MP): original handcraft prompts from LAMA
  - Fine-tuning(FT)
  - MP with FT: fine-tuning language model with handcraft prompts
  - P-tuning: continuous prompts

3. Experiments

LAMA dataset

Prompt type	Model	P@1
Original (MP)	BERT-base	31.1
	BERT-large	32.3
	E-BERT	36.2
Discrete	LPAQA (BERT-base)	34.1
	LPAQA (BERT-large)	39.4
	AutoPrompt (BERT-base)	43.3
P-tuning	BERT-base	48.3
	BERT-large	**50.6**

Model	MP	FT	MP+FT	P-tuning
BERT-base (109M)	31.7	51.6	52.1	52.3 (+20.6)
-AutoPrompt (Shin et al., 2020)	-	-	-	45.2
BERT-large (335M)	33.5	54.0	55.0	54.6 (+21.1)
RoBERTa-base (125M)	18.4	49.2	50.0	49.3 (+30.9)
-AutoPrompt (Shin et al., 2020)	-	-	-	40.0
RoBERTa-large (355M)	22.1	52.3	52.4	53.5 (+31.4)
GPT2-medium (345M)	20.3	41.9	38.2	46.5 (+26.2)
GPT2-xl (1.5B)	22.8	44.9	46.5	54.4 (+31.6)
MegatronLM (11B)	23.1	OOM*	OOM*	**64.2** (+41.1)

* MegatronLM (11B) is too large for effective fine-tuning.

Table 2. Knowledge probing Precision@1 on LAMA-34k (left) and LAMA-29k (right). P-tuning outperforms all the discrete prompt searching baselines. And interestingly, despite fixed pre-trained model parameters, P-tuning overwhelms the fine-tuning GPTs in LAMA-29k. (MP: Manual prompt; FT: Fine-tuning; MP+FT: Manual prompt augmented fine-tuning; PT: P-tuning).

MP: 사람이 만든 prompt
Discrete: 이산적으로 생성된 prompt
P-tuning: continuous prompt

3. Experiments

SuperGlue dataset

- Tasks (7 dataset for NLU tasks evaluation)
  - BOOlQ, MultiRC: question and answering
  - CB, RTE: textual entailment
  - WiC: co-reference resolution
  - COPA: causal reasoning
  - WSC: word sense disambiguation

- Dataset
  - Fully-supervised setting
  - Few-shot setting: train/dev set 32개

3. Experiments

SuperGlue dataset

Fully-supervised learning on SuperGLUE dev with base-scale models

Method	BoolQ (Acc.)	CB (Acc.)	CB (F1)	WiC (Acc.)	RTE (Acc.)	MultiRC (EM)	MultiRC (F1a)	WSC (Acc.)	COPA (Acc.)	Avg.
BERT-base-cased (109M)										
Fine-tuning	72.9	85.1	73.9	71.1	68.4	16.2	66.3	63.5	67.0	66.2
MP zero-shot	59.1	41.1	19.4	49.8	54.5	0.4	0.9	62.5	65.0	46.0
MP fine-tuning	73.7	87.5	90.8	67.9	70.4	13.7	62.5	60.6	70.0	67.1
P-tuning	73.9	89.2	92.1	68.8	71.1	14.8	63.3	63.5	72.0	68.4
GPT2-base (117M)										
Fine-tune	71.2	78.6	55.8	65.5	67.8	17.4	65.8	63.0	64.4	63.0
MP zero-shot	61.3	44.6	33.3	54.1	49.5	2.2	23.8	62.5	58.0	48.2
MP fine-tuning	74.8	87.5	88.1	68.0	70.0	23.5	69.7	66.3	78.0	70.2
P-tuning	75.0 (+1.1)	91.1 (+1.9)	93.2 (+1.1)	68.3 (-2.8)	70.8 (-0.3)	23.5 (+7.3)	69.8 (+3.5)	63.5 (+0.0)	76.0 (+4.0)	70.4 (+2.0)

Table 3. Fully-supervised learning on SuperGLUE dev with base-scale models. MP refers to manual prompt. For a fair comparison, MP zero-shot and MP fine-tuning report results of a single pattern, while anchors for P-tuning are selected from the same prompt. Subscript in red represents advantages of GPT with P-tuning over the best results of BERT.

3. Experiments

SuperGlue dataset

Fully-supervised learning on SuperGLUE dev with large-scale models

Method	BoolQ (Acc.)	CB (F1)	CB (Acc.)	WiC (Acc.)	RTE (Acc.)	MultiRC (EM)	MultiRC (F1a)	WSC (Acc.)	COPA (Acc.)	Avg.
BERT-large-cased (335M)										
Fine-tune*	77.7	94.6	93.7	74.9	75.8	24.7	70.5	68.3	69.0	72.5
MP zero-shot	49.7	50.0	34.2	50.0	49.9	0.6	6.5	61.5	58.0	45.0
MP fine-tuning	77.2	91.1	93.5	70.5	73.6	17.7	67.0	80.8	75.0	73.1
P-tuning	77.8	96.4	97.4	72.7	75.5	17.1	65.6	81.7	76.0	74.6
GPT2-medium (345M)										
Fine-tune	71.0	73.2	51.2	65.2	72.2	19.2	65.8	62.5	66.0	63.1
MP zero-shot	56.3	44.6	26.6	54.1	51.3	2.2	32.5	63.5	53.0	47.3
MP fine-tuning	78.3	96.4	97.4	70.4	72.6	32.1	74.4	73.0	80.0	74.9
P-tuning	78.9 (+1.1)	98.2 (+1.8)	98.7 (+1.3)	69.4 (-5.5)	75.5 (-0.3)	29.3 (+4.6)	74.2 (+3.7)	74.0 (-7.7)	81.0 (+5.0)	75.6 (+1.0)

* We report the same results taken from SuperGLUE (Wang et al., 2019b).

Table 4. Fully-supervised learning on SuperGLUE dev with large-scale models. MP refers to manual prompt. For fair comparison, MP zero-shot and MP fine-tuning report results of a single pattern, while anchors for P-tuning are selected from the same prompt. Subscripts in red represents improvements of GPT with P-tuning over the best results of BERT.

3. Experiments

SuperGlue dataset

Few-shot learning (32 train samples) on SuperGLUE dev

Dev size	Method	BoolQ (Acc.)	CB (Acc.)	CB (F1)	WiC (Acc.)	RTE (Acc.)	MultiRC (EM)	MultiRC (F1a)	WSC (Acc.)	COPA (Acc.)
32	PET*	73.2±3.1	82.9±4.3	74.8±9.2	51.8±2.7	62.1±5.3	33.6±3.2	74.5±1.2	79.8±3.5	85.3±5.1
	PET best†	75.1	86.9	83.5	52.6	65.7	35.2	75.0	80.4	83.3
	P-tuning	77.8 (+4.6)	92.9 (+10.0)	92.3 (+17.5)	56.3 (+4.5)	76.5 (+14.4)	36.1 (+2.5)	75.0 (+0.5)	84.6 (+4.8)	87.0 (+1.7)
Full	GPT-3	77.5	82.1	57.2	55.3	72.9	32.5	74.8	75.0	92.0
	PET‡	79.4	85.1	59.4	52.4	69.8	37.9	77.3	80.1	95.0
	iPET§	80.6	92.9	92.4	52.2	74.0	33.0	74.0	-	-

* We report the average and standard deviation of each candidate prompt's average performance.
† We report the best performed prompt selected on *full* dev dataset among all candidate prompts.
‡ With additional ensemble and distillation.
§ With additional data augmentation, ensemble, distillation and self-training.

Table 5. Few-shot learning (32 train samples) on SuperGLUE dev. Previous few-shot learning approaches use the original full dev set ($\mathcal{D}_{dev}$) for validation, which does not make sense. We construct a new dev set ($\mathcal{D}_{dev32}$) with 32 unused samples from original training set. Under fair comparison, P-tuning significantly outperforms PET ($\mathcal{D}_{dev32}$) and PET best ($\mathcal{D}_{dev32}$) on all tasks. More interestingly, P-tuning even outperforms GPT-3, PET ($\mathcal{D}_{dev}$) and iPET ($\mathcal{D}_{dev}$) on 4 out of 7 tasks. Subscripts in red represents the improvements of P-tuning over PET($\mathcal{D}_{dev32}$).

4. Conclusion

- P-tuning 을 통해 GPT 가 NLU task 에서 BERT 만큼 경쟁력이 있음을 보임
- P-tuning 을 사용했을 때, few-shot 및 fully-supervised setting 모든 설정에서

  GPT가 BERT 의 성능 능가
- LAMA 및 SuperGlue dataset 에서 SOTA 달성

  => P-tuning 을 사용했을 때 언어모델이 사전훈련동안 더 많은 지식을 학습
- P-tuning 을 사용했을 때 BERT 성능도 향상됨

참고 자료

- DSBA 세미나 prompt-based learning http://dsba.korea.ac.kr/seminar/?mod=document&uid=1829
- seopbo.log, GPT Understands, Too https://velog.io/@seopbo/GPT-Understands-Too
- 열쩡강쥐 블로그, GPT Understands, Too 리뷰 https://hye-z.tistory.com/18?category=501949
- SOOFTWARE 블로그, Sooftware NLP - P-tuning Paper Review https://sooftware.io/p_tuning/
- 끄적끄적 블로그, [2021] GPT Understands, Too (P-tuning) https://soundprovider.tistory.com/entry/2021-GPT-Understands-Too-P-tuning
- P-tuning code [github]
- hryang 블로그, NLP 이해하기 https://hryang06.github.io/nlp/NLP/
- Quora, "Are neural networks continuous?" https://www.quora.com/Are-neural-networks-continuous