LLM FineTuning

Peft-LoRA 방법론을 활용해서
학교 안내챗봇 만들기

30407 문태서 - 선린 응용프로그램 발표

프로젝트 주제와 목적

학교생활 도움 챗봇

- 학생들에게 정확한 정보를 전달
- 보다 다양한 형식의 답변도 처리 가능
- 자연스러운 대화 및 이해 가능
- 개인화된 응답 가능
- 지속적인 학습 및 업데이트 가능
- 사용자 경험 향상
- 다양한 언어 및 표현 지원

프로젝트 구성

- 베이스 LLM 모델: Gemma-2B-it
- 학습 데이터셋: 자체제작 데이터셋(학교 정보)
- 파인튜닝 학습방식: PEFT-LoRA
- UI 및 사용환경 개선: Gradio

LLM의 정의와 특징

Large Language Model

- 대규모 텍스트 데이터를 학습한 NLP 분야의 모델
- 텍스트 생성, 이해, 번역, 요약, 감정분석 등 다양한 형식의 자연어 처리
- 문장의 전후 맥락을 고려하여 답변을 생성

- DNN 아키텍처 기반
- 트랜스포머 구조를 활용한 모델들이 다수

LLM의 작동방식

전체적인 프로세스

- 1. 토큰화: 입력 프롬프트를 토큰화
- 2. 벡터화: 각 토큰을 고유한 다차원 벡터로 변환 (Word2vec..)
- 3. 추론: 각 벡터 데이터들 간의 유사도를 계산
- 4. 생성: 찾은 학습 데이터를 기반으로 새로운 텍스트 생성 (토큰으로 반환)
- 5. 디코딩: 반환받은 토큰을 다시 자연어로 임베딩

- 토크나이저: 텍스트를 토큰으로 분활하는 도구
- 임베딩 테이블: 토큰을 벡터로 매핑하는 데이블

LLM의 작동방식

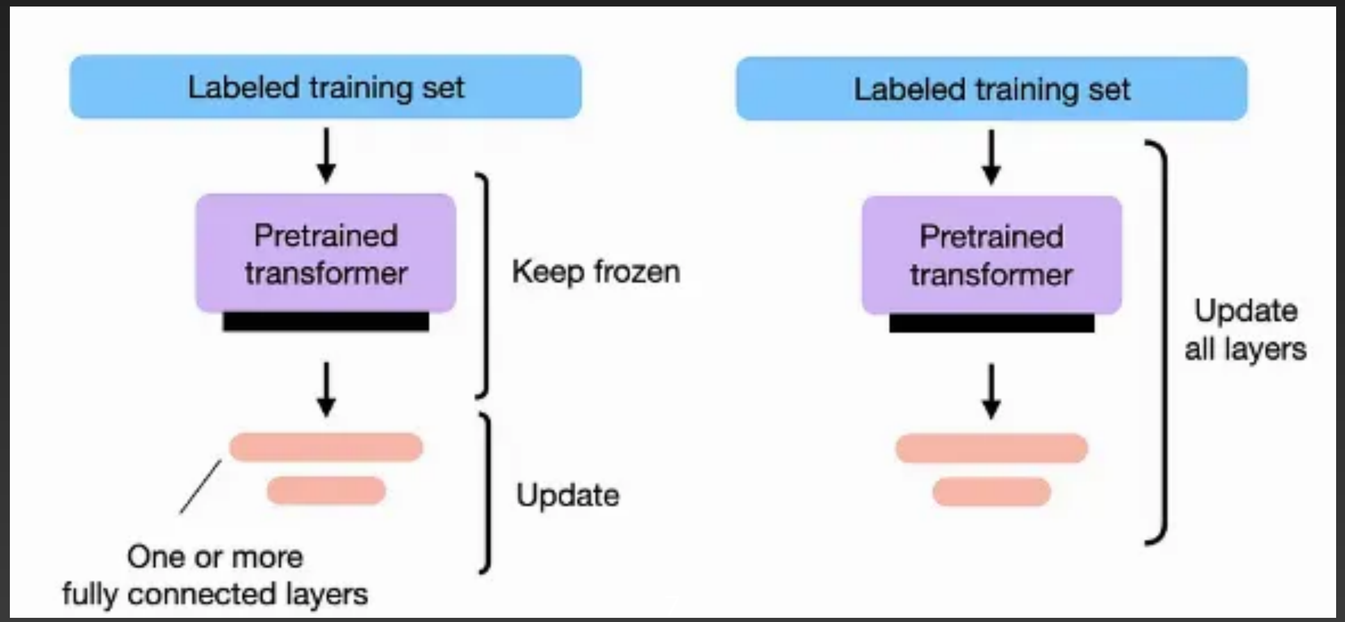
1. 프롬프트 토큰화

- 토큰: LLM 모델 추론을 위해 사용되는 기본 단위
- 토큰화: 자연어 문장을 특정 기준으로 나눔
  - 예시:
  - “Hello. nice to meet you” => [“Hello”, “.”, “nice”, “to”, “meet”, “you”]
- 토큰화의 기준
  - 1. 공백기준
  - 2. 구두점기준

파인튜닝 정의

파인튜닝 방식

- 사전 학습된 모델의 매개 변수를 업데이트하는 프로세스
  - Full Fine-tuning: 모든 파라미터를 재조정
  - Pepurposing: 일부 파라미터만 조정 / 추가적인 모듈을 도입

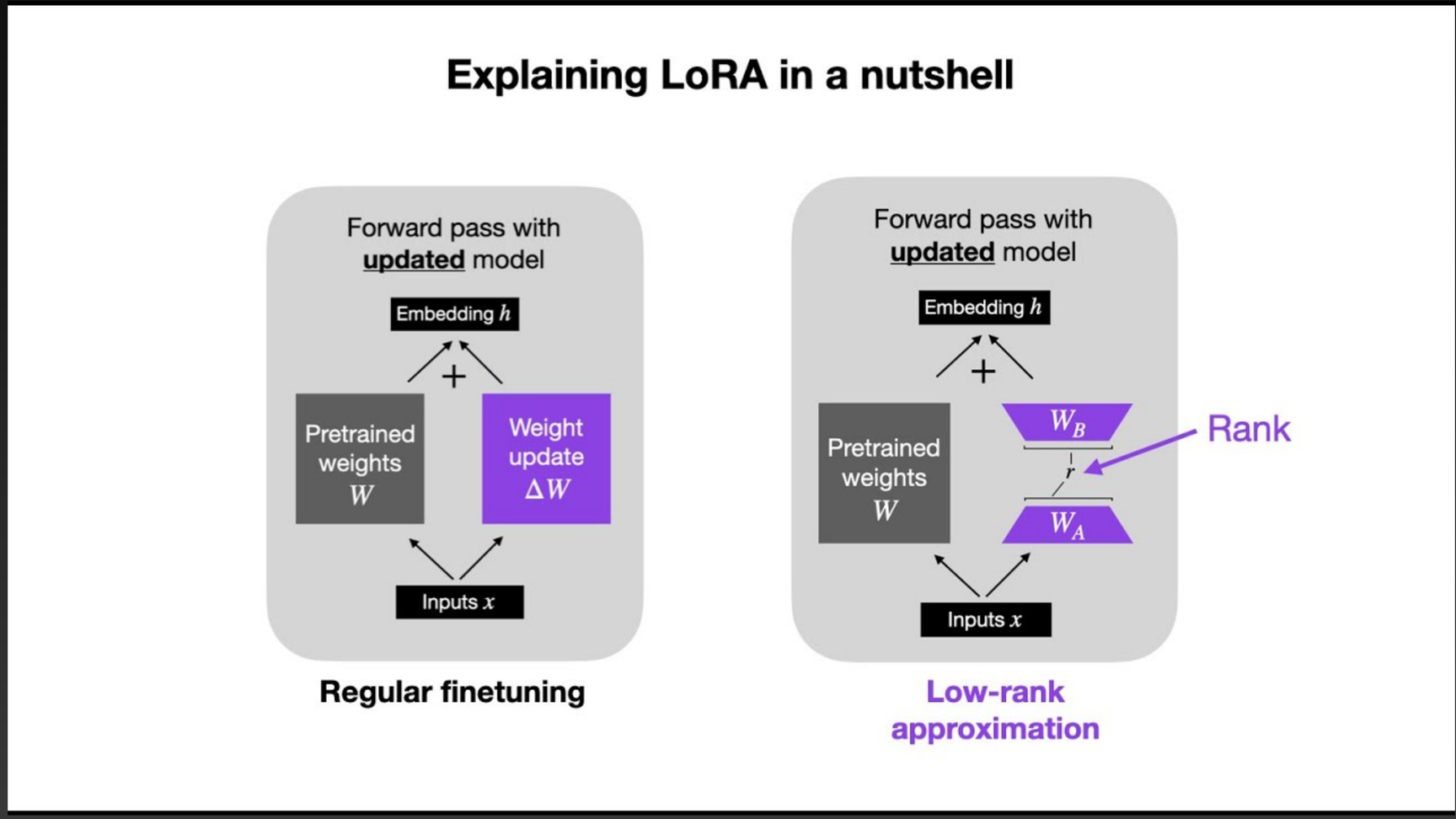


LLM 모델의 파인튜닝

만약 GPT-3 175B 모델을 파인튜닝 해야한다면?

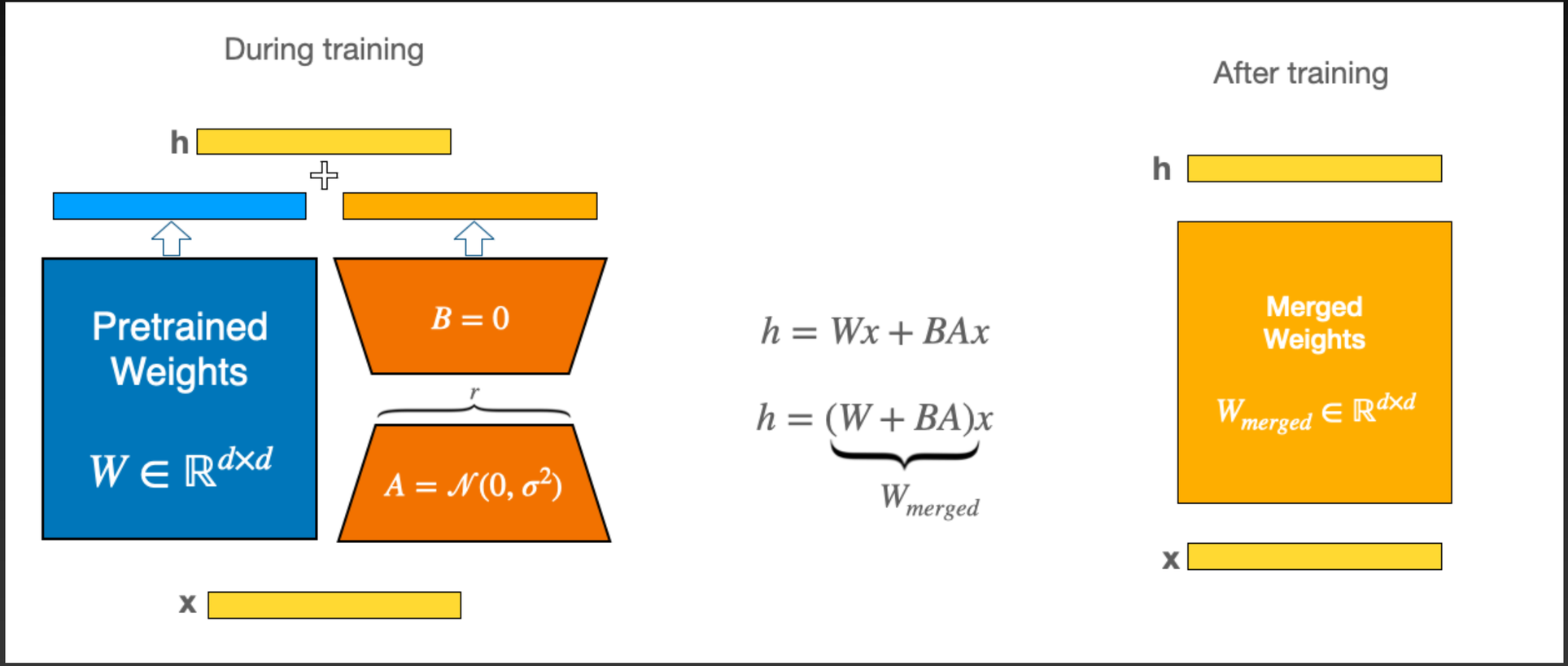
조정해야 될 파라미터 : 1750억 개

PEFT-LoRA 방법론
Low Rank Adapters



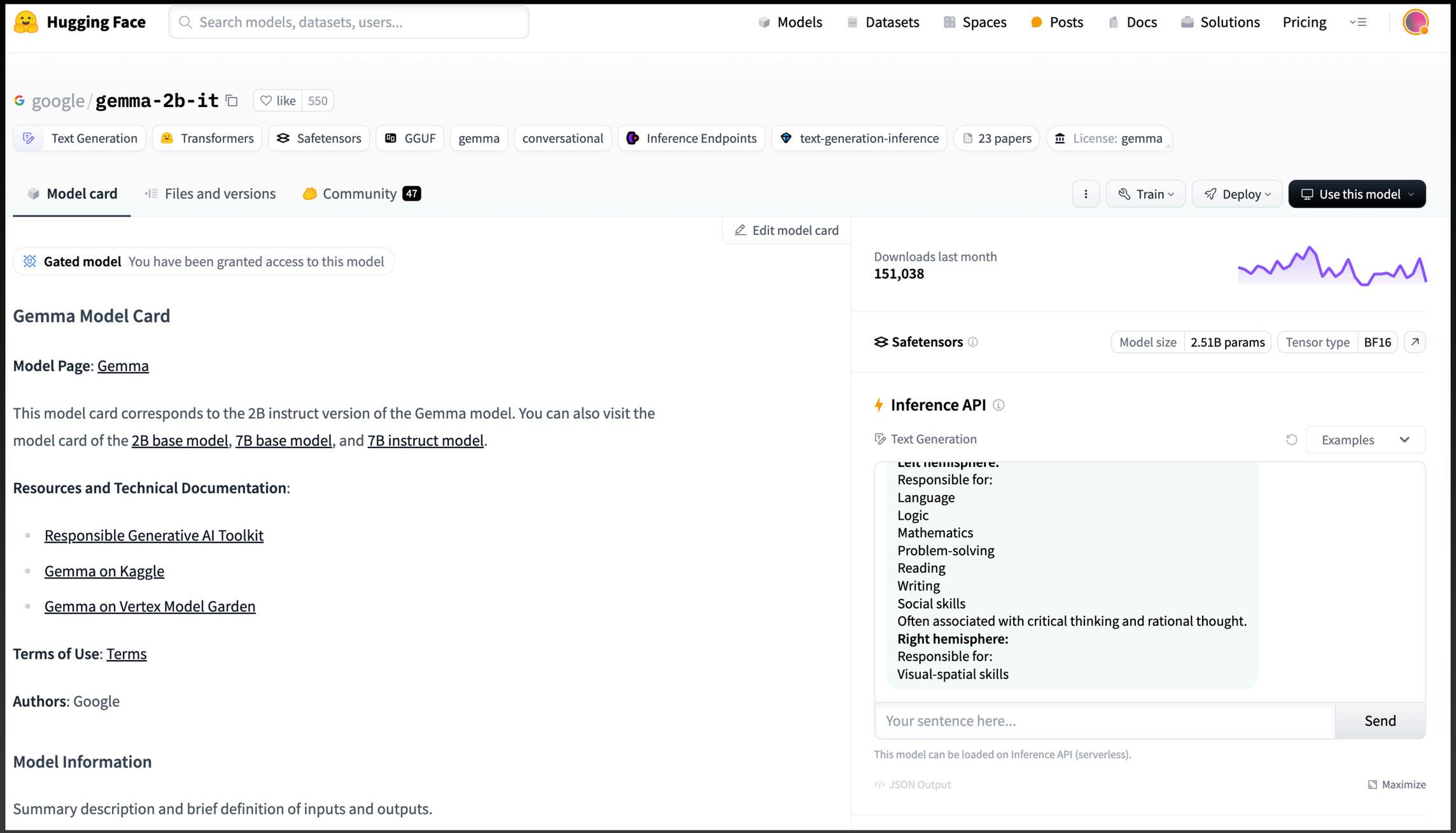
Merged LoRA Model

Low Rank Adapters



Gemma-2b-it 모델

- Google에서 공개한 오픈 llm 모델
- 파라미터 개수: 20억 개
- Instruction Tuning된 모델


Hugging Face
Search models, datasets, users...
Models
Datasets
Spaces
Posts
Docs
Solutions
Pricing
google / gemma-2b-it
like 550
Text Generation
Transformers
Safetensors
GGUF
gemma
conversational
Inference Endpoints
text-generation-inference
23 papers
License: gemma
Model card
Files and versions
Community 47
Train
Deploy
Use this model
Edit model card
Gated model You have been granted access to this model
Gemma Model Card
Model Page: Gemma
This model card corresponds to the 2B instruct version of the Gemma model. You can also visit the model card of the 2B base model, 7B base model, and 7B instruct model.
Resources and Technical Documentation:
Responsible Generative AI Toolkit
Gemma on Kaggle
Gemma on Vertex Model Garden
Terms of Use: Terms
Authors: Google
Model Information
Summary description and brief definition of inputs and outputs.
Downloads last month
151,038
Safetensors
Model size 2.51B params
Tensor type BF16
Inference API
Text Generation
Examples
Responsible for:
Language
Logic
Mathematics
Problem-solving
Reading
Writing
Social skills
Often associated with critical thinking and rational thought.
Right hemisphere:
Responsible for:
Visual-spatial skills
Your sentence here...
Send
This model can be loaded on Inference API (serverless).
JSON Output
Maximize

CAPABILITY	BENCHMARK	DESCRIPTION	Gemma 7B	Llama-2 7B	Llama-2 13B
General	MMLU 5-shot, top-1	Representation of questions in 57 subjects (incl. STEM, humanities and others)	64.3	45.3	54.8
Reasoning	BBH -	Diverse set of challenging tasks requiring multi-step reasoning	55.1	32.6	39.4
	HellaSwag 0-shot	Commonsense reasoning for everyday tasks	81.2	77.2	80.7
Math	GSM8K maj@1	Basic arithmetic manipulations (incl. Grade School math problems)	46.4	14.6	28.7
	MATH 4-shot	Challenging math problems (incl. algebra, geometry, pre-calculus, and others)	24.3	2.5	3.9
Code	HumanEval pass@1	Python code generation	32.3	12.8	18.3

데이터셋 구성

학교 정보를 담은 데이터셋 제작

- 데이터셋 형식: Parquet

- 데이터셋 구조:
- 데이터 개수:

```
Dataset({
    features: ['instruction', 'output'],
    num_rows: 128
})
```

Datasets: taeseo06 / **SchoolHelperChatbot-dataset** like 0

Tags: Croissant

Dataset card | Viewer | Files and versions | Community 1 | Settings

Dataset Viewer

Auto-converted to Parquet | API | View in Dataset Viewer

Split (1)
train · 128 rows

Search this dataset

instruction string · *lengths* 9 38	output string · *lengths* 22 233
선린인터넷고등학교는 어느 동네에 위치해있어?	선린인터넷고등학교는 용산구 남영동에 위치해있습니다.
선린인터넷고등학교의 위치를 알려줘	선린인터넷고등학교는 용산구 원효로 97길에 위치해있습니다.
학교의 동네를 알려줘	선린인터넷고등학교는 서울시 용산구 남영동에 위치해있습니다.
선린인터넷고등학교의 자세한 위치를 알려줘	선린인터넷고등학교는 서울특별시 용산구 원효로 97길 33-4에 위치해있습니다.
선린인터넷고등학교의 자세한 위치를 알고싶어	선린인터넷고등학교의 자세한 주소는 서울특별시 용산구 원효로 97길 33-4 입니다.
남영역에서 선린인터넷고등학교 까지 거리가 어떻게 돼?	남영역에서 선린인터넷고등학교 까지 도보로 약 7분 입니다. 남영역 출구 바로 건너편에 선린인터넷고등학교가 자리하고있으며, 횡단보도를 2번 건너면 선린인터넷고등학교 정문이 나옵니…
선린인터넷고등학교 주변에는 뭐가있어?	선린인터넷고등학교 주변에는 숙명여자대학교, 선린중학교, 남영역 등이 있습니다.
선린인터넷고등학교 주변에 숙명여자대학교가 있어?	네, 선린인터넷고등학교 주변에는 숙명여자대학교가 있습니다. 정확하게는, 선린 중학교 후문 건너편 지하주차장을 건너면 숙명여자대학교 캠퍼스가 나옵니다. 도보로 약 7분 정도가 소요…

Previous 1 2 Next

Downloads last month 5

Use in Datasets library | Add dataset card

Size of downloaded dataset files: 12.1 kB

Size of the auto-converted Parquet files: 12.1 kB

Number of rows: 128

Split (1)

train · 128 rows

Search this dataset

instruction string · *lengths*  9 38	output string · *lengths*  22 233
선린인터넷고등학교는 어느 동네에 위치해있어?	선린인터넷고등학교는 용산구 남영동에 위치해있습니다.
선린인터넷고등학교의 위치를 알려줘	선린인터넷고등학교는 용산구 원효로 97길에 위치해있습니다.
학교의 동네를 알려줘	선린인터넷고등학교는 서울시 용산구 남영동에 위치해있습니다.
선린인터넷고등학교의 자세한 위치를 알려줘	선린인터넷고등학교는 서울특별시 용산구 원효로 97길 33-4에 위치해있습니다.
선린인터넷고등학교의 자세한 위치를 알고싶어	선린인터넷고등학교의 자세한 주소는 서울특별시 용산구 원효로 97길 33-4 입니다.

1호관은 어디에 있어?
1호관은 어떻게 갈 수 있어?
1호관으로 가는 길을 알려줘

후문에서 선린중학교 운동장을 따라 돌면, 이동하는 방향 그대로 2호관으로 들어갈 수 있습니다.
1호관은 운동장과 맞닿아 있으며, 정문에서 교내로 올라오는 길 오른쪽에 위치해 있습니다.
1호관은 운동장 맞은편에 위치해 있으며, 2호관과 연결되어 있습니다. 따라서 운동장 오른쪽에 위치한 외부 출입문으로 들어가거나, 2호관과 연결된 통로를 통해서 이동할 수 있습니다.
1호관은 운동장 맞은편에 위치해 있으며, 2호관과 연결되어 있습니다. 따라서 운동장 오른쪽에 위치한 외부 출입문으로 들어가거나, 2호관과 연결된 통로를 통해서 이동할 수 있습니다.

LoRA 설정

```
##########################################
# 6-1. LoRA config 정의

config = LoraConfig(
    r=32,
    lora_alpha=16,
    target_modules=["gate_proj", "down_proj", "up_proj"],
    lora_dropout=0.05,
    bias="none",
    task_type="CAUSAL_LM")
```

파인튜닝 과정

```
########################################
# 7-2. Trainer class 정의
trainer = transformers.Trainer(
        model=model,
        train_dataset=train_data,
        eval_dataset=tokenized_datasets,

        args=transformers.TrainingArguments( # 훈련에 이용될 하이퍼파라미터
            per_device_train_batch_size = 4,
            gradient_accumulation_steps = 4,
            warmup_ratio=0.06,
            num_train_epochs=25,
            learning_rate=1e-4, #
            fp16=True,
            logging_steps=1,
            optim="adamw_torch",
            evaluation_strategy="epoch", # 얼리스탑핑 도입을 위해 epoch마다 평가하는 설정  # 기존 값 = "no" - 6/10
            save_strategy="epoch",
            max_grad_norm = 1.0,
            save_steps = 30,
            lr_scheduler_type='cosine',
            output_dir= './custom_LLM',
            save_total_limit=2,
            # load_best_model_at_end=False, # 비활성화 - 6/10
            load_best_model_at_end=True,  # 최적의 모델을 로드 - 6/10 (위 코드와 상반)
            ddp_find_unused_parameters=False,
            group_by_length = False,
        ),
        data_collator=transformers.DataCollatorForSeq2Seq(
            tokenizer, pad_to_multiple_of=8, return_tensors="pt", padding=True
        ),
        callbacks=[EarlyStoppingCallback(early_stopping_patience=3)], # 조기 종단 콜백 추가
        # early_stopping_patience=3  # 조기 중단 설정 - 6/10 (얼리스탑핑 옵션 도입: 과적합 방지),

    )

model.config.use_cache = False
model.print_trainable_parameters() # 훈련하는 파라미터의 % 체크

if torch.__version__ >= "2" and sys.platform != "win32":
    model = torch.compile(model)
```

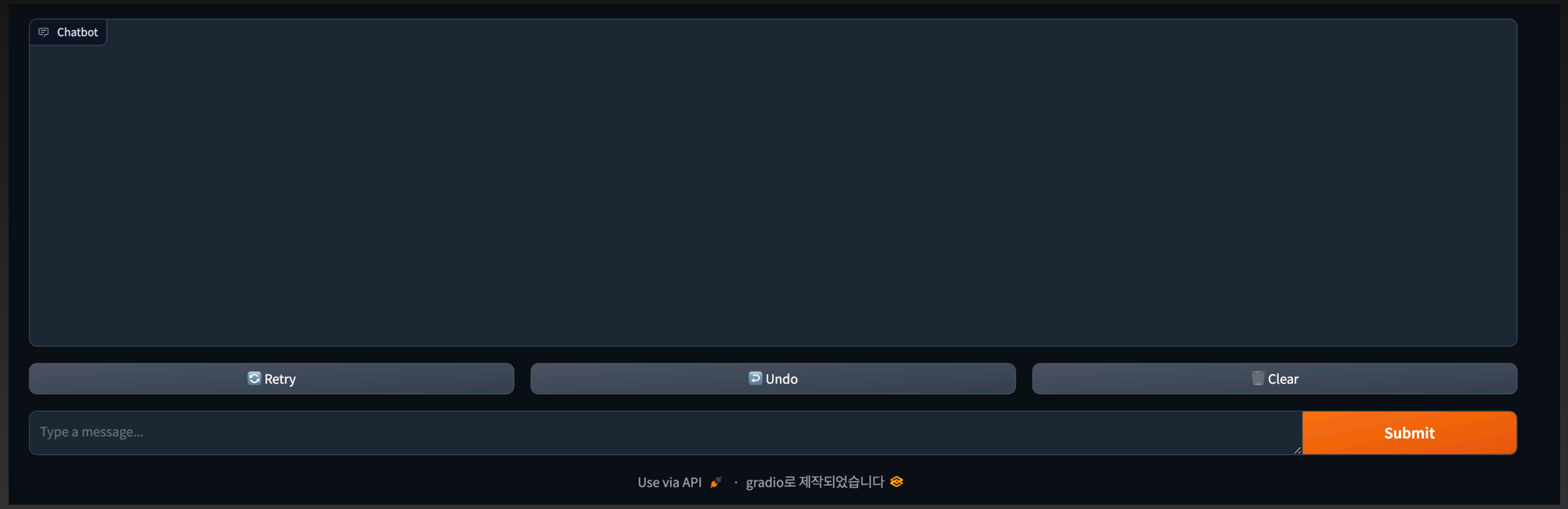
파인튜닝 과정

- 과적합 방지를 위해 early_stopping 도입 (epoch)
- KFold 교차검증 도입

Gradio 챗봇 UI

```python
# 6/7 테스트 성공 (영어, 한국어)

# import
from transformers import AutoTokenizer, AutoModelForCausalLM
import torch
import re
import gradio as gr

# 허깅페이스 API 토큰 지정
hf_token = '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' # 이곳에 허깅페이스 토큰을 삽입

# 모델 지정 및 데이터타입 설정 (16bit)
model_id = "taeseo06/SchoolHelperChatbot-Gemma2B-Finetuning" # model: https://huggingface.co/datasets/taeseo06/SchoolHelperChatbot-dataset
dtype = torch.bfloat16

# 채팅 함수
def gemma_chat(message, history):
    tokenizer = AutoTokenizer.from_pretrained(model_id, use_auth_token=hf_token)
    model = AutoModelForCausalLM.from_pretrained(
        model_id,
        use_auth_token=hf_token,
        torch_dtype=dtype,
    )

    device = torch.device("cuda" if torch.cuda.is_available() else "cpu")
    model.to(device)

    chat = [
        { "role": "user", "content": message },
    ]

    prompt = tokenizer.apply_chat_template(chat, tokenize=False, add_generation_prompt=True)

    inputs = tokenizer.encode(prompt, add_special_tokens=False, return_tensors="pt").to(device)
    outputs = model.generate(input_ids=inputs, max_new_tokens=2048)

    response = tokenizer.decode(outputs[0])

    # 응답을 정리
    response_cleaned = re.split("model", response)

    # 응답 반환
    return response_cleaned[1]

# Gradio 인터페이스 생성 및 실행
gr.ChatInterface(gemma_chat).launch(debug=True)
```

챗봇 UI 구성


Chatbot
Retry
Undo
Clear
Type a message...
Submit
Use via API
· gradio로 제작되었습니다

시연

개선점

질의응답