

预训练大模型技术实践

以 ChatGLM-6B 为例

ChatGLM-6B: 单卡版本开源的对话模型



你好



你好！我是人工智能助手 ChatGLM，很高兴见到你，欢迎问我任何问题。

① 以上内容为 AI 生成，不代表开发者立场，请勿删除或修改本标记



帮我写一个介绍ChatGLM的博客提纲



好的，以下是一个可能的博客提纲：

I. 前言

1. 简要介绍 ChatGLM 的背景和主要功能

II. ChatGLM 的特点

1. 智能化的问答和对话功能

2. 自定义主题和语言

3. 丰富的知识库和语言模型

4. 支持多种平台和语言

III. ChatGLM 的应用场景

1. 在教育、医疗、金融等领域的应用

2. 帮助人们更高效地获取信息和解决问题

3. 推动人工智能的发展和普及

IV. ChatGLM 的未来发展方向

1. 进一步完善智能化和个性化服务

2. 拓展应用场景和用户规模

3. 推动人工智能在各个领域的应用和创新

- **充分的中英双语预训练：** ChatGLM2-6B 在 1:1 比例的中英语料上训练了 1.4T 的 token 量，兼具双语能力，相比于 ChatGLM-6B 初代模型，性能大幅提升。
- **较低的部署门槛：** FP16 半精度下，ChatGLM-6B 需要至少 13GB 的显存进行推理，结合模型量化技术，这一需求可以进一步降低到 10GB (INT8) 和 6GB (INT4)，使得 ChatGLM-6B 可以部署在消费级显卡上。
- **更长的序列长度：** 相比 GLM-10B (序列长度 1024)，ChatGLM-6B 序列长度达 2048，ChatGLM2-6B 序列长度达 8192，支持更长对话和应用。
- **人类意图对齐训练：** 使用了监督微调、反馈自助、人类反馈强化学习等方式，使模型初具理解人类指令意图的能力。

ChatGLM-6B vs ChatGLM2-6B

	ChatGLM-6B	ChatGLM2-6B
最大上下文长度	2048	8192
推理速度	32字符/秒	45字符/秒 (42%↑)
训练量	1T token	1.4T token
最小部署显存/文本长度	6G / 1k	6G / 8K

英文综合能力 +23% (MMLU)

Model	Average
ChatGLM-6B	40.63
ChatGLM2-6B (base)	47.86
ChatGLM2-6B	45.46

中文综合能力 +33% (C-Eval)

Model	Average
ChatGLM-6B	38.9
ChatGLM2-6B (base)	51.7
ChatGLM2-6B	50.1

数学能力 +571% (GSM-8k)

Model	Accuracy
ChatGLM-6B	4.82
ChatGLM2-6B (base)	32.37
ChatGLM2-6B	28.05

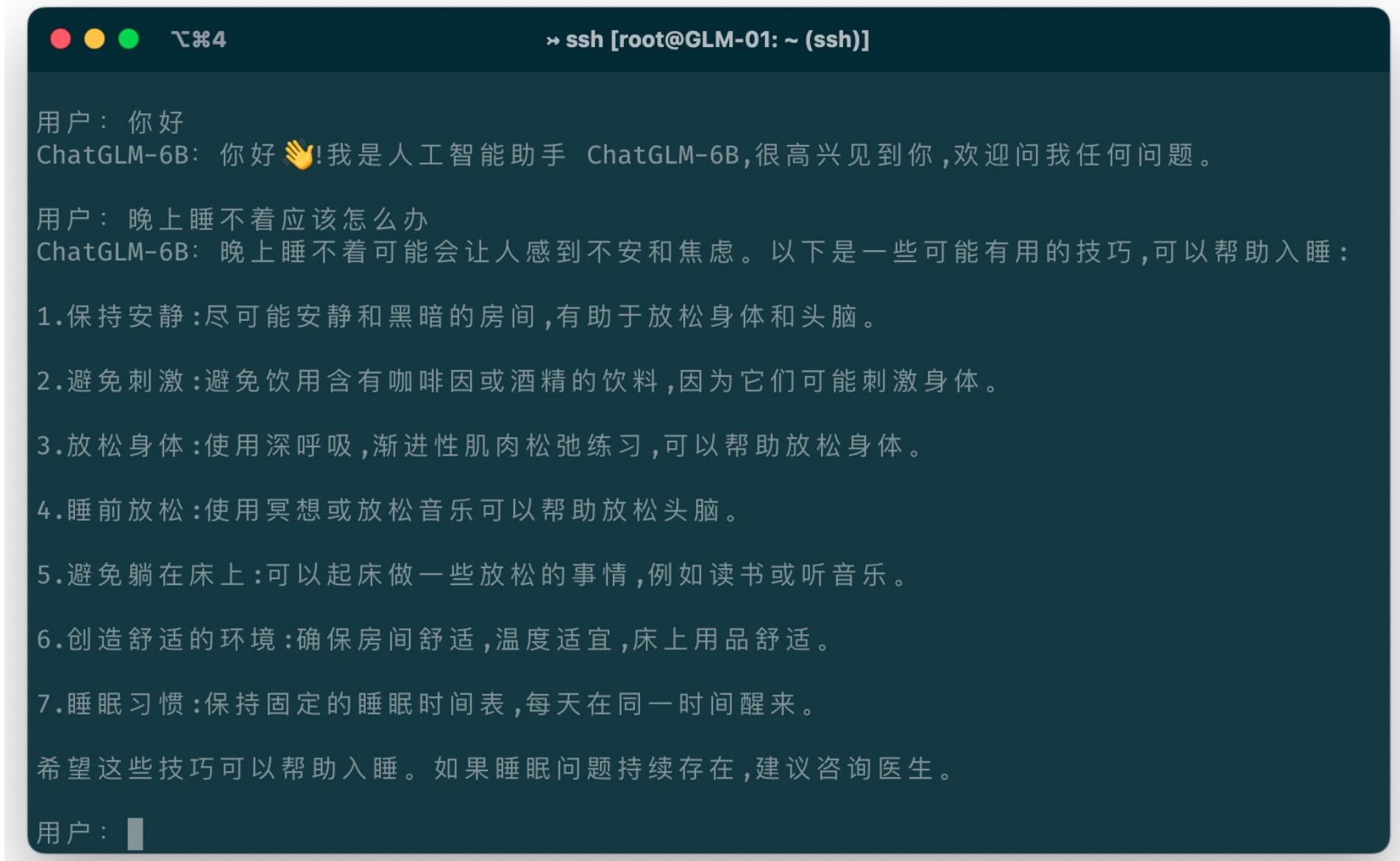
| 预训练大模型技术实践

- 以 ChatGLM-6B 为例
- **模型本地部署：**
 - 本地单机运行ChatGLM-6B模型进行对话
- **模型微调：**
 - 有限资源下对ChatGLM-6B模型进行高效微调

ChatGLM-6B 本地推理部署

ChatGLM-6B / ChatGLM2-6B

ChatGLM2-6B：本地运行Demo



ChatGLM

Chatbot

Input...

Clear History

Maximum length 2048

Top P 0.7

Temperature 0.95

Submit

The screenshot shows the ChatGLM user interface. At the top left is a 'Chatbot' button. Below it is a large input field with placeholder text 'Input...'. To the right of the input field are three sliders: 'Maximum length' set to 2048, 'Top P' set to 0.7, and 'Temperature' set to 0.95. Above the sliders is a 'Clear History' button. At the bottom of the interface is a large orange 'Submit' button.

ChatGLM2-6B 本地部署demo

ChatGLM2-6B 部署: 基础环境配置

1) 设备和基础环境

- 1 - python
- 2 - pytorch
- 3 - Nvidia 显卡 (更快)
- 4 - Windows CPU / Mac (速度较慢)

2) python依赖安装

```
1 git clone https://github.com/THUDM/ChatGLM2-6B
2 cd ChatGLM2-6B
3 pip install -r requirements.txt
```

protobuf
transformers==4.30.2
cpm_kernels
torch>=2.0
gradio
mdtex2html
sentencepiece
accelerate

ChatGLM2-6B 部署: 模型权重准备

代码直接调用 (正常访问 huggingface)

```
>>> from transformers import AutoTokenizer, AutoModel  
>>> tokenizer = AutoTokenizer.from_pretrained("THUDM/chatglm2-6b", trust_remote_code=True)  
>>> model = AutoModel.from_pretrained("THUDM/chatglm2-6b", trust_remote_code=True, device='cuda')  
>>> model = model.eval()
```

```
>>> response, history = model.chat(tokenizer, "你好", history=[])  
>>> print(response)
```

你好👋!我是人工智能助手 ChatGLM2-6B,很高兴见到你,欢迎问我任何问题。

```
>>> response, history = model.chat(tokenizer, "晚上睡不着应该怎么办", history=history)
```

```
>>> print(response)
```

晚上睡不着可能会让你感到焦虑或不舒服,但以下是一些可以帮助你入睡的方法:

1. 制定规律的睡眠时间表:保持规律的睡眠时间表可以帮助你建立健康的睡眠习惯,使你更容易入睡。尽量在每天的相同时间上床,并在同一时间起床。
2. 创造一个舒适的睡眠环境:确保睡眠环境舒适,安静,黑暗且温度适宜。可以使用舒适的床上用品,并保持房间通风。
3. 放松身心:在睡前做些放松的活动,例如泡个热水澡,听些轻柔的音乐,阅读一些有趣的书籍等,有助于缓解紧张和焦虑,使你更容易入睡。
4. 避免饮用含有咖啡因的饮料:咖啡因是一种刺激性物质,会影响你的睡眠质量。尽量避免在睡前饮用含有咖啡因的饮料,例如咖啡,茶和可乐。
5. 避免在床上做与睡眠无关的事情:在床上做些与睡眠无关的事情,例如看电影,玩游戏或工作等,可能会干扰你的睡眠。
6. 尝试呼吸技巧:深呼吸是一种放松技巧,可以帮助你缓解紧张和焦虑,使你更容易入睡。试着慢慢吸气,保持几秒钟,然后缓慢呼气。

如果这些方法无法帮助你入睡,你可以考虑咨询医生或睡眠专家,寻求进一步的建议。

ChatGLM2-6B 部署: 模型权重准备

代码直接调用 (正常访问 huggingface)

```
>>> from transformers import AutoTokenizer, AutoModel
>>> tokenizer = AutoTokenizer.from_pretrained("THUDM/chatglm2-6b", trust_remote_code=True)
>>> model = AutoModel.from_pretrained("THUDM/chatglm2-6b", trust_remote_code=True, device='cuda')
>>> model = model.eval()
>>> response, history = model.chat(tokenizer, "你好", history=[])
>>> print(response)
你好👋!我是人工智能助手 ChatGLM2-6B,很高兴见到你,欢迎问我任何问题。
>>> response, history = model.chat(tokenizer, "晚上睡不着应该怎么办", history=history)
>>> print(response)
```

晚上睡不着可能会让你感到焦虑或不舒服,但以下是一些可以帮助你入睡的方法:

1. 制定规律的睡眠时间表:保持规律的睡眠时间表可以帮助你建立健康的睡眠习惯,使你更容易入睡。尽量在每天的相同时间上床,并在同一时间起床。
2. 创造一个舒适的睡眠环境:确保睡眠环境舒适,安静,黑暗且温度适宜。可以使用舒适的床上用品,并保持房间通风。
3. 放松身心:在睡前做些放松的活动,例如泡个热水澡,听些轻柔的音乐,阅读一些有趣的书籍等,有助于缓解紧张和焦虑,使你更容易入睡。
4. 避免饮用含有咖啡因的饮料:咖啡因是一种刺激性物质,会影响你的睡眠质量。尽量避免在睡前饮用含有咖啡因的饮料,例如咖啡,茶和可乐。
5. 避免在床上做与睡眠无关的事情:在床上做些与睡眠无关的事情,例如看电影,玩游戏或工作等,可能会干扰你的睡眠。
6. 尝试呼吸技巧:深呼吸是一种放松技巧,可以帮助你缓解紧张和焦虑,使你更容易入睡。试着慢慢吸气,保持几秒钟,然后缓慢呼气。

如果这些方法无法帮助你入睡,你可以考虑咨询医生或睡眠专家,寻求进一步的建议。

ChatGLM2-6B 部署: 模型权重准备

从本地加载模型

- 1) 从 Huggingface 上下载模型实现

```
GIT_LFS_SKIP_SMUDGE=1 git clone https://huggingface.co/THUDM/chatglm2-6b
# Cloning into 'chatglm2-6b'...
# remote: Enumerating objects: 124, done.
# remote: Counting objects: 100% (48/48), done.
# remote: Compressing objects: 100% (38/38), done.
```

- 2) 从[清华云盘](#)下载模型参数文件，放到本地`chatglm2-6b`下

- 3) 将模型加载路径换成本地`chatglm2-6b`

```
# 原始加载 code
tokenizer = AutoTokenizer.from_pretrained("THUDM/chatglm2-6b", trust_remote_code=True)
model = AutoModel.from_pretrained("THUDM/chatglm2-6b", trust_remote_code=True).cuda()

# 更换为本地加载 code
tokenizer = AutoTokenizer.from_pretrained("YOUR_PATH_TO_chatglm2-6b", trust_remote_code=True)
model = AutoModel.from_pretrained("YOUR_PATH_TO_chatglm2-6b", trust_remote_code=True).cuda()
```



chatglm-6b



当前路径: chatglm-6b

	名称 ▲
□	ice_text.model
□	LICENSE
□	MODEL_LICENSE
□	pytorch_model-00001-of-00008.bin
□	pytorch_model-00002-of-00008.bin
□	pytorch_model-00003-of-00008.bin
□	pytorch_model-00004-of-00008.bin
□	pytorch_model-00005-of-00008.bin
□	pytorch_model-00006-of-00008.bin
□	pytorch_model-00007-of-00008.bin
□	pytorch_model-00008-of-00008.bin

ChatGLM2-6B 部署: 模型权重准备

CPU 部署

如果你没有 GPU 硬件的话，也可以在 CPU 上进行推理，但是推理速度会更慢。使用方法如下（需要大概 32GB 内存）

```
model = AutoModel.from_pretrained("THUDM/chatglm2-6b", trust_remote_code=True).float()
```

如果你的内存不足的话，也可以使用量化后的模型

```
model = AutoModel.from_pretrained("THUDM/chatglm2-6b-int4", trust_remote_code=True).float()
```

在 cpu 上运行量化后的模型需要安装 `gcc` 与 `openmp`。多数 Linux 发行版默认已安装。对于 Windows，可在安装 [TDM-GCC](#) 时勾选 `openmp`。Windows 测试环境 `gcc` 版本为 `TDM-GCC 10.3.0`，Linux 为 `gcc 11.3.0`。

更多细节可以参见：
<https://github.com/THUDM/ChatGLM2-6B>

Mac 部署

对于搭载了 Apple Silicon 或者 AMD GPU 的 Mac，可以使用 MPS 后端来在 GPU 上运行 ChatGLM2-6B。需要参考 Apple 的 [官方说明](#) 安装 PyTorch-Nightly（正确的版本号应该是`2.x.x.dev2023xxxx`，而不是 `2.x.x`）。

目前在 MacOS 上只支持[从本地加载模型](#)。将代码中的模型加载改为从本地加载，并使用 mps 后端：

```
model = AutoModel.from_pretrained("your local path", trust_remote_code=True).to('mps')
```



本地Web-Demo

```
# 运行仓库中的 web_demo.py  
python web_demo.py
```

```
# 在浏览器中进入对应地址
```

更多细节可以参见：
<https://github.com/THUDM/ChatGLM2-6B>



ChatGLM

Chatbot

Input...

Clear History

Maximum length 2048

Top P 0.7

Temperature 0.95

Submit

A screenshot of the ChatGLM web application. At the top, it says "ChatGLM". Below that is a "Chatbot" button. The main area has an "Input..." placeholder. To the right are three sliders: "Maximum length" set to 2048, "Top P" set to 0.7, and "Temperature" set to 0.95. A large orange "Submit" button is at the bottom.

ChatGLM2-6B 部署: 推理参数设置

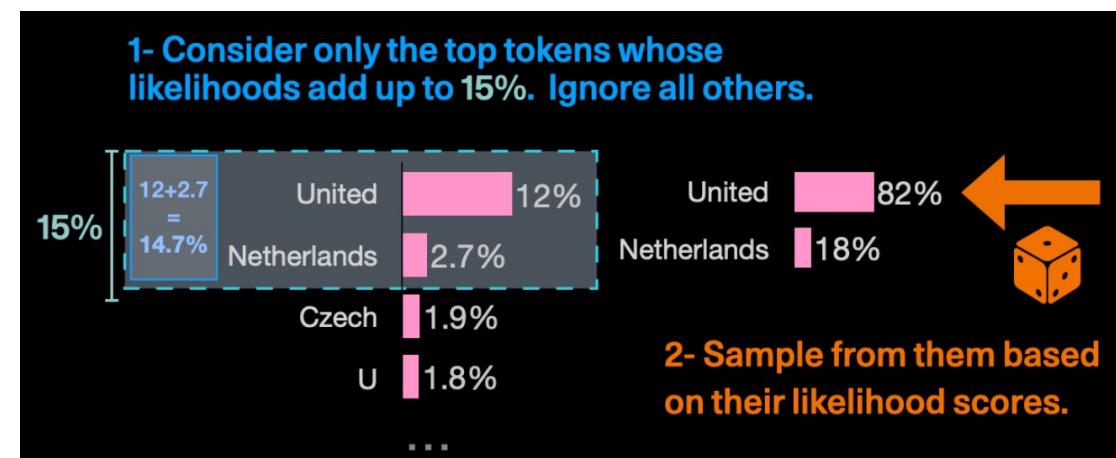
```
# 推理参数
`temperature`: 采样温度, [ 0.0, 1.0 ];
  - 越大回答多样性越高; 推荐0.95
`top_p`: nucleus采样阈值, [ 0.0, 1.0 ];
  - 越大回答多样性越高; 推荐0.7
`max_length`: 最大输出长度
```

- **temperature, top_p**数值越小, 随机性越小;
- 最优值对于不同任务或经过微调后可能发生变化

temperature: 锐化概率分布

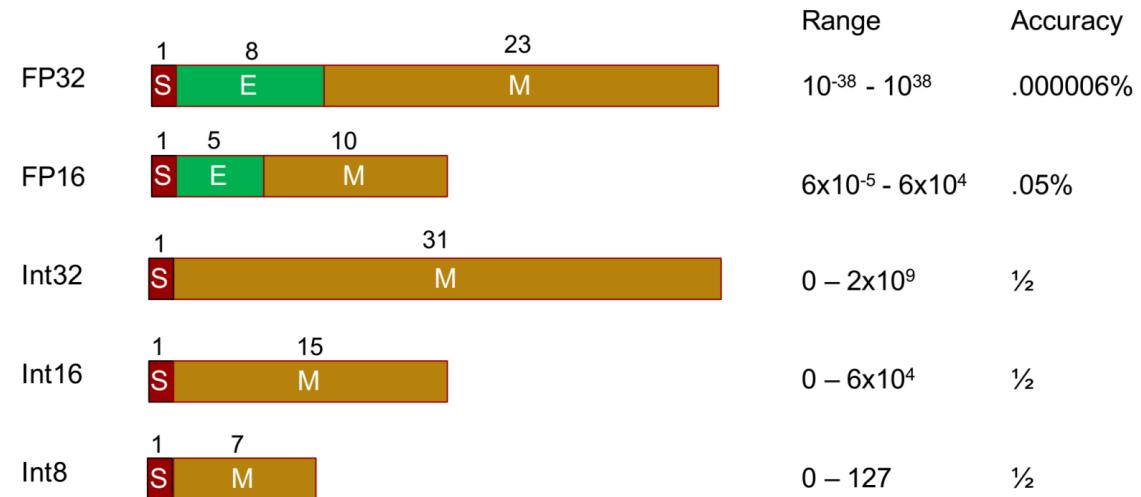
$$P(x_i|x_{1:i-1}) = \frac{\exp(u_i/t)}{\sum_j \exp(u_j/t)}$$

top_p: 根据概率对候选做截断



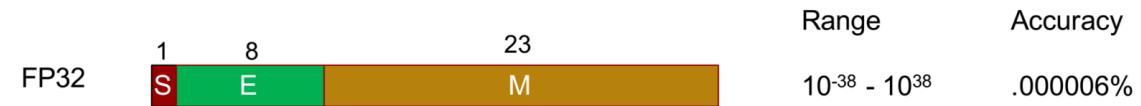
ChatGLM2-6B 部署: 量化

W4/8A16量化方案，显著节省显存：
存储时INT4/8，计算时FP16



精度	ChatGLM-6B	硬件要求 (最低)
FP16	13GB	
INT8	8GB	任意8GB显存以上GPU * 1
INT4	6GB	

ChatGLM2-6B 部署: 量化



W4 低成本部署

存储 模型量化

默认情况下，模型以 FP16 精度加载，运行上述代码需要大概 13GB 显存。如果你的 GPU 显存有限，可以尝试以量化方式加载模型，使用方法如下：

```
# 按需修改，目前只支持 4/8 bit 量化  
model = AutoModel.from_pretrained("THUDM/chatglm2-6b", trust_remote_code=True).quantize(8).cuda()
```

模型量化会带来一定的性能损失，经过测试，ChatGLM2-6B 在 4-bit 量化下仍然能够进行自然流畅的生成。

如果你的内存不足，可以直接加载量化后的模型：

```
model = AutoModel.from_pretrained("THUDM/chatglm2-6b-int4", trust_remote_code=True).cuda()
```

ChatGLM-6B 微调

ChatGLM-6B 微调

- 模型微调：

- 利用新的数据，对已有模型进行继续训练

- 为什么做模型微调？

- 增强模型通用和人类对其能力
 - 模型更加适应专业化领域场景
 - 法律、金融、医疗、教育等场景

ChatGLM-6B 微调

- Repo

<https://github.com/THUDM/ChatGLM-6B>

Features

- 基于 P-tuning v2 的参数高效微调
- 全参数微调
- 支持多轮对话数据集和生成任务数据集

| 大模型微调范式

- **Prompting**

手写 prompt 来引导模型适应下游任务。存在问题：成本高，难以达到最优

Prompt	P@1
[X] is located in [Y]. (<i>original</i>)	31.29
[X] is located in which country or state? [Y].	19.78
[X] is located in which country? [Y].	31.40
[X] is located in which country? In [Y].	51.08

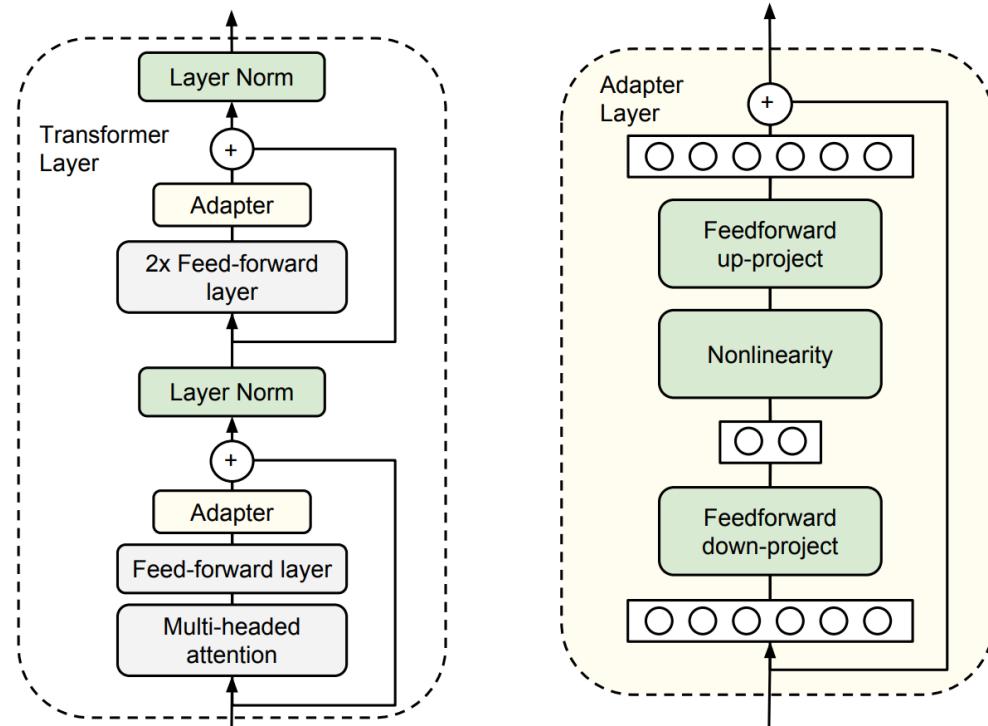
- **解决方法**

Auto search of Discrete prompts

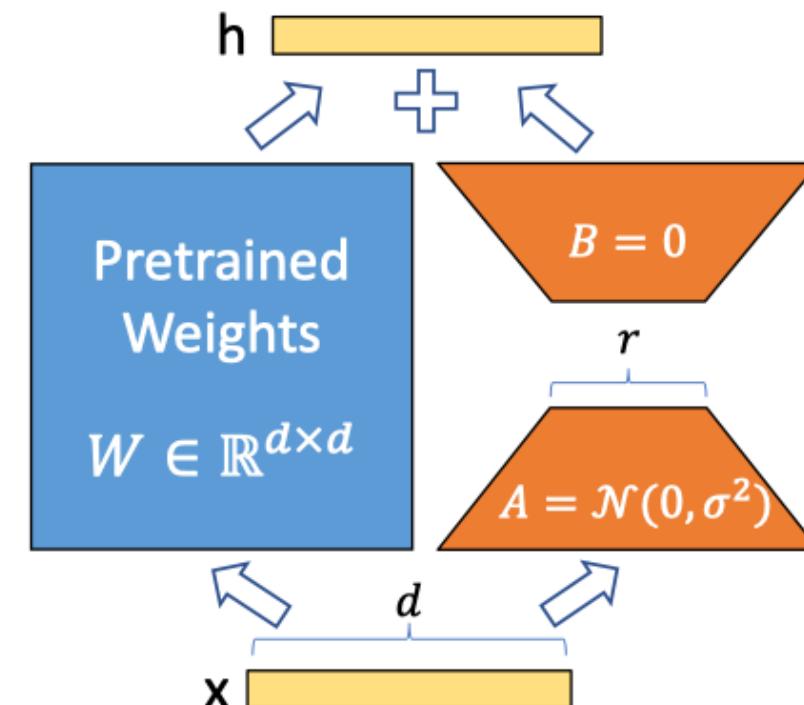
Optimization of continuous prompts

大模型微调范式

- Pre-training:
 - 全量训练，无标注数据
- Fine-tuning:
 - 全量训练，标注数据
- Parameter efficient tuning
 - Adapter: 加入adapter层训练，会引入额外推理开销；
 - Prompt/Prefix tuning: 效果更优，需要调参；
 - Lora: 依靠权重的低秩分解特点，没有额外推理开销；



Adapter: 插入额外可训练层

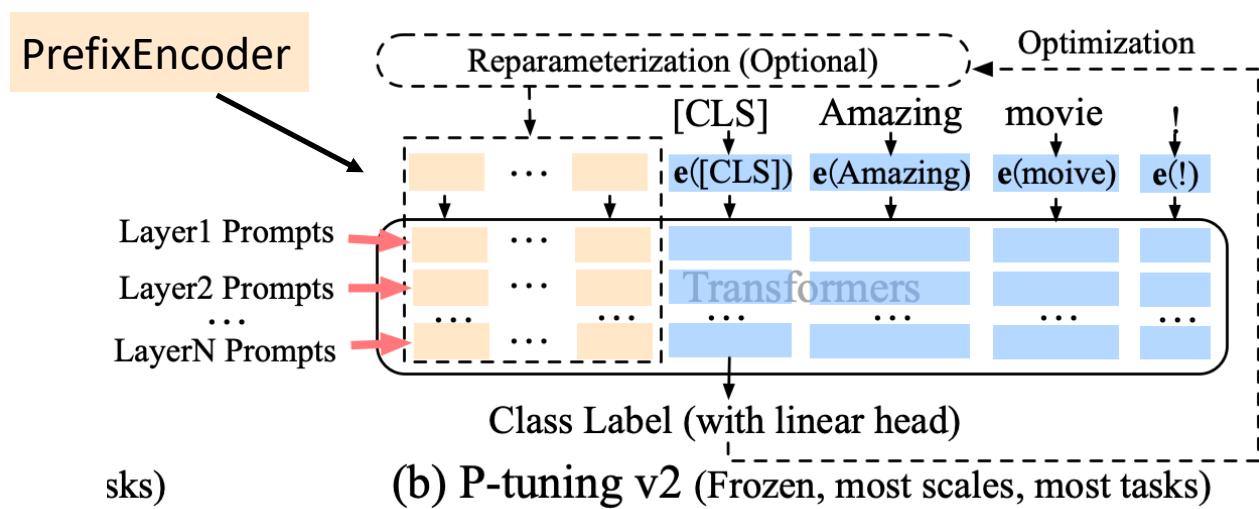
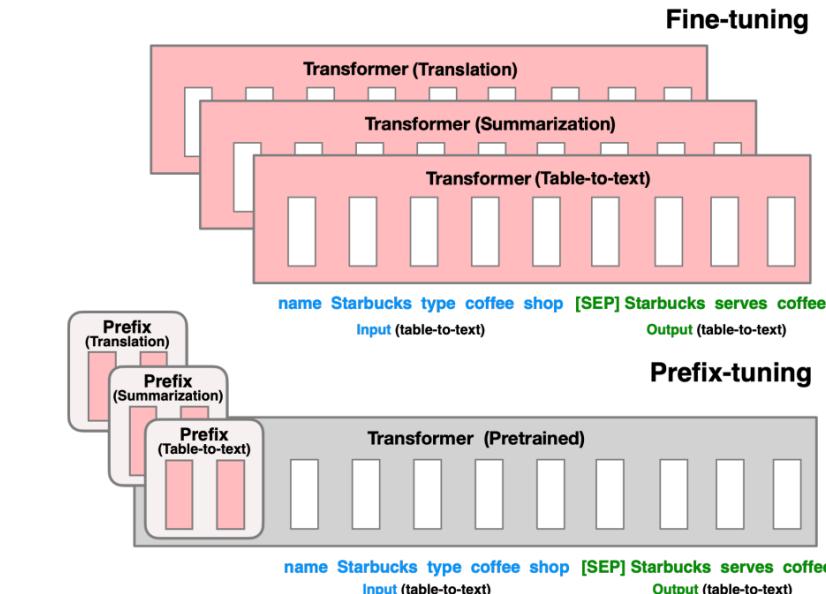


LoRA: 仅训练A、B, $W = W_0 + BA$

ChatGLM-6B 微调: P-tuning v2 原理

• PrefixEncoder

- 每一层前面插入一段 prompt
- 模型微调参数降低到原来的 0.1%-0.5%
- **冻结模型参数**, 只训练每层的 prompt
- **降低训练所需显存**
 - 只需要保存和载入 PrefixEncoder
 - 模型保存空间小



[Prefix-Tuning: Optimizing Continuous Prompts for Generation \(Li & Liang, ACL-IJCNLP 2021\)](#)

[P-Tuning: Prompt Tuning Can Be Comparable to Fine-tuning Across Scales and Tasks \(Liu et al., ACL 2022\)](#)

微调ChatGLM-6B: 模型权重准备

- 自动下载模型权重:

```
from transformers import AutoTokenizer, AutoModel
tokenizer = AutoTokenizer.from_pretrained("THUDM/chatglm-6b", trust_remote_code=True)
model = AutoModel.from_pretrained("THUDM/chatglm-6b", trust_remote_code=True).half().cuda()
```

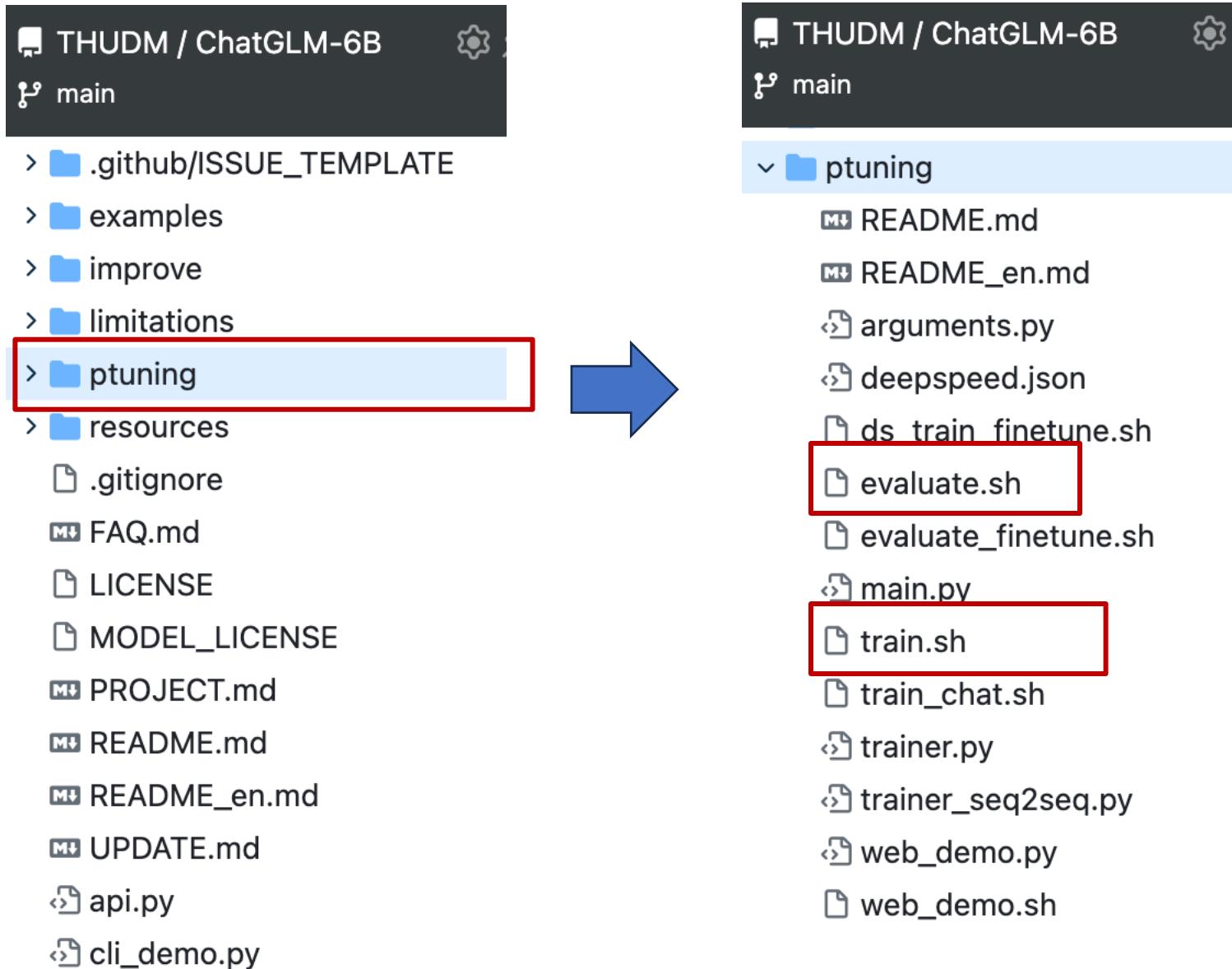
- 手动下载模型权重:

<https://cloud.tsinghua.edu.cn/d/674208019e314311ab5c>

- 手动下载模型实现 (包括Tokenizer) :

```
GIT_LFS_SKIP_SMUDGE=1 git clone https://huggingface.co/THUDM/chatglm-6b
```

微调ChatGLM-6B : P-tuning v2 微调



微调ChatGLM-6B : P-tuning v2 微调

- 命令: `bash train.sh`

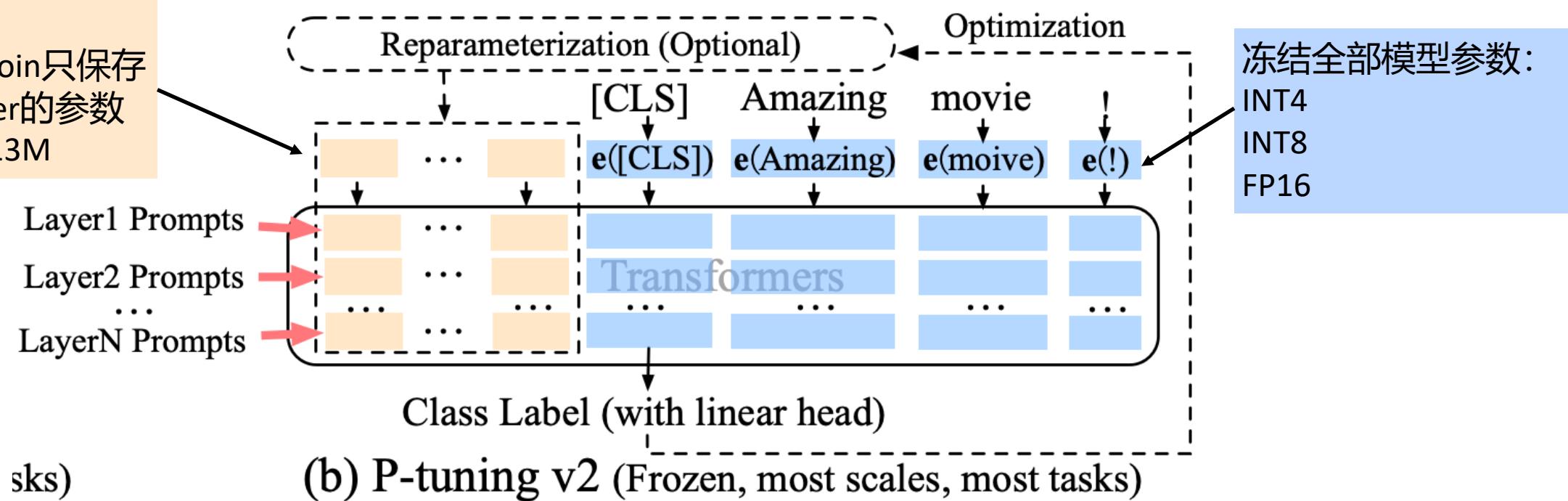
```
CUDA_VISIBLE_DEVICES=0 python3 main.py \
--do_train \
--train_file AdvertiseGen/train.json \
--validation_file AdvertiseGen/dev.json \
--prompt_column content \
--response_column summary \
--overwrite_cache \
--model_name_or_path THUDM/chatglm-6b \
--output_dir output/adgen-chatglm-6b-pt \
--overwrite_output_dir \
--max_source_length 64 \
--max_target_length 64 \
--per_device_train_batch_size 16 \
--per_device_eval_batch_size 1 \
--gradient_accumulation_steps 1 \
--predict_with_generate \
--max_steps 3000 \
--logging_steps 10 \
--save_steps 1000 \
--learning_rate 2e-2 \
--pre_seq_len 128 \
--quantization_bit 4
```

数据集相关
可以设为本地模型路径
微调模型保存路径
Batch size
总训练表数
打印loss
保存checkpoint
学习率, p-tuning用的lr较大
Prompt长度, 不设置时全参微调
量化设置

微调ChatGLM-6B : P-tuning v2 微调

可训练参数: FP32

模型Checkpoint只保存
PrefixEncoder的参数
: 3.8G -> 113M



```
quantization_bit=4 per_device_train_batch_size=1 gradient_accumulation_steps=16
```

INT4的模型参数被冻结，一次训练迭代会以1的批处理大小进行16次累加的前后向传播，等效为16的总批处理大小，此时最低只需**6.7G显存**。

微调ChatGLM-6B: 全参数 Finetune

- 需要安装 **deepspeed**, 需要多卡跑
- 使用较小的学习率 (1e-5, 1e-4)
- 命令: `bash ds_train_finetune.sh`

```
MASTER_PORT=$(shuf -n 1 -i 10000-65535)
```

硬件要求	ChatGLM-6B
推理	任意8GB显存以上GPU * 1
微调	A100 (80G) * 4 A100 (40G) * 8 V100 (32G) * 12

```
deepspeed --num_gpus=4 --master_port $MASTER_PORT main.py \
--deepspeed deepspeed.json \
...
--per_device_train_batch_size 4 \
--gradient_accumulation_steps 1 \
...
--learning_rate 1e-4 \
--fp16
```

微调ChatGLM-6B: 推理

命令

```
bash evaluate.sh
```

```
CUDA_VISIBLE_DEVICES=0 python3 main.py \
--do_predict \
--validation_file AdvertiseGen/dev.json \
--test_file AdvertiseGen/dev.json \
--overwrite_cache \
--prompt_column content \
--response_column summary \
--model_name_or_path THUDM/chatglm-6b \
--ptuning_checkpoint ./output/adgen-chatglm-6b-pt-128-2e-2/checkpoint-3000 \
--output_dir ./output/adgen-chatglm-6b-pt-128-2e-2 \
--overwrite_output_dir \
--max_source_length 256 \
--max_target_length 256 \
--per_device_eval_batch_size 1 \
--predict_with_generate \
--pre_seq_len 128 \
--quantization_bit 4
```

微调ChatGLM-6B: 示例数据集

ADGEN (广告生成)

```
{  
    "content": "类型#上衣*版型#宽松*版型#显瘦*图案#线条*衣样式#衬衫*衣袖型#泡泡袖*衣款式#抽绳",  
    "summary": "这件衬衫的款式非常的宽松, 利落的线条可以很好的隐藏身材上的小缺点, 穿在身上有着很好的显瘦效果。领口装饰了一个可爱的抽绳, 漂亮的绳结展现出了十足的个性, 配合时尚的泡泡袖型, 尽显女性甜美可爱的气息。"  
}
```

• **Output[微调前]:** 这件上衣的材质是牛仔布,颜色是白色,风格是简约,图案是刺绣,衣样式是外套,衣款式是破洞。

• **Output[微调后]:** 这是一款简约的牛仔外套,破洞设计,将牛仔布破洞,带来一种随意与个性。破洞的牛仔外套,展现出时尚气息,带来一种休闲感。同时,刺绣图案,让整件外套更加立体。

微调ChatGLM-6B: 示例数据集

ADGEN (广告生成)

```
{  
    "content": "类型#裙*版型#显瘦*风格#文艺*风格#简约*图案#印花*图案#撞色*裙下摆#压褶*裙长#连衣裙*裙领型#圆领",  
    "summary": "文艺个性的印花连衣裙,藏青色底蕴,低调又大气,撞色太阳花分布整个裙身,绚丽而美好,带来时尚减龄的气质。基础款的舒适圆领,简约不失大方,勾勒精致脸庞。领后是一粒包布扣固定,穿脱十分方便。前片立体的打褶设计,搭配后片压褶的做工,增添层次和空间感,显瘦又有型。"  
}
```

• **Output[微调前]:** 1. 连衣裙: 简约风格, 裙长为膝盖以上, 裙领型为圆领。2. 裙下摆: 压褶设计, 使裙摆呈现出流畅的褶皱效果。3. 裙领型: 裙领型为圆领, 使穿上连衣裙后更加有型。4. 版型: 采用显瘦设计, 让连衣裙看起来更加苗条。5. 风格: 文艺风格, 让连衣裙更加有内涵和品味。6. 图案: 印花设计, 在连衣裙上印有独特的图案。7. 撞色: 采用撞色设计, 让连衣裙在色彩上更加鲜明、富有层次感。

• **Output[微调后]:** 这是一款文艺范的连衣裙, 以印花为元素, 采用简约的印花, 既能够突出文艺气质, 又能够展现简约风。在印花的同时又有领子和裙摆的压褶设计, 更加凸显文艺气质。简约而不会过于单调, 搭配出街, 穿着十分舒适。

微调ChatGLM-6B: 示例数据集

ADGEN (广告生成)

```
{  
    "content": "类型#上衣*版型#宽松*版型#显瘦*图案#线条*衣样式#衬衫*衣袖型#泡泡袖*衣款式#抽绳",  
    "summary": "这件衬衫的款式非常的宽松，利落的线条可以很好的隐藏身材上的小缺点，穿在身上有着很好的显瘦效果。领口装饰了一个可爱的抽绳，漂亮的绳结展现出了十足的个性，配合时尚的泡泡袖型，尽显女性甜美可爱的气息。"  
}
```

Model	BART	Fine-tune (fp16)	P-tuning v2 (INT4, l=128)	LoRA
BLEU-4	10.2	8.01	8.10	7.62
Rouge-1		31.23	31.12	30.60
Rouge-2		7.36	7.11	6.96
Rouge-L		25.08	24.97	24.80
Training Loss		3.00	3.74	3.32

微调ChatGLM-6B: 多轮对话数据集

可以提供聊天历史，然后在训练命令中指定 `--history_column`

```
{  
    "prompt": "是的。上下水管都好的",  
    "response": "那就要检查线路了，一般风扇继电器是由电脑控制吸合的，如果电路存在断路，或者电脑坏了的话会出现继电器不吸合的情况！",  
    "history": [  
        [  
            "长城h3风扇不转。继电器好的。保险丝好的传感器新的风扇也新的这是为什么。就是继电器缺一个信号线",  
            "用电脑能读数据流吗？水温多少"  
        ],  
        [  
            "95",  
            "上下水管温差怎么样啊？空气是不是都排干净了呢？"  
        ]  
    ]  
}
```

微调ChatGLM-6B: 多轮对话数据集

代码会自动把聊天历史拼接成以下形式

- Input

[Round 0]

问:长城h3风扇不转。继电器好的。保险丝好的传感器新的风扇也新的这是为什么。就是继电器缺一个信号线

答:用电脑能读数据流吗?水温多少

[Round 1]

问:95

答:上下水管温差怎么样啊?空气是不是都排干净了呢?

[Round 2]

问:是的。上下水管都好的

答:

- Label

那就要检查线路了,一般风扇继电器是由电脑控制吸合的,如果电路存在断路,或者电脑坏了的话会出现继电器不吸合的情况!

要注意超过输入长度 `max_source_length` 的内容会被截,需要注意轮数

**行业大模型微调
PowerLawGLM法律模型**

PowerLawGLM

通用大模型：

近日，一名律师在一起法庭案件中，依赖聊天机器人 ChatGPT 进行法律文献检索（legal research），导致提交了错误的信息。这一事件揭示了人工智能在法律领域的潜在风险，包括误传错误信息。

- Pro：具有强大语言理解和生成能力
- Con：专业知识及数据的相对欠缺，往往会有法律知识错误、专业引用偏差、法律体系差异等问题。

PowerLawGLM：

- 聚焦于法律细分领域，针对中文法律场景的应用效果具有独特优势，具备丰富的法律知识和法律语言理解能力。
- 基于GLM-130B

PowerLawGLM 示例



女友不和我结婚了，我给她的彩礼能要回来吗？



彩礼是在婚姻关系中产生的一种特殊的财物，通常是男方为了与女方结婚而向女方家庭支付的金钱或其他财物。如果女方拒绝结婚，彩礼的返还问题通常是由双方协商解决，如果无法协商，可以向法院提起诉讼。

在中国，法律规定彩礼应当是自愿的、合法的、合理的。如果彩礼是用于共同生活，那么女方有权拒绝返还彩礼。但是，如果彩礼是用于个人消费或赌博等非法行为，那么女方有权要求返还彩礼。

ChatGLM



女友不和我结婚了，我给她的彩礼能要回来吗？



根据法律规定，如果你们尚未办理结婚登记手续，你可以请求对方返还彩礼。根据《最高人民法院关于适用<中华人民共和国民法典>婚姻家庭编的解释(一)》第五条的规定，如果查明属于双方未办理结婚登记手续、双方办理结婚登记手续但确未共同生活、婚前给付并导致给付人生活困难这些情形，人民法院应当支持当事人返还彩礼的请求。

PowerLawGLM

PowerLawGLM 训练

第一步——阅读大量法律文本。

高质量法律文本数据（裁判文书、法律法规、法律知识问答等）模型增量训练

第二步——与法律对话场景对齐，具备法律场景的对话能力。

法律领域对话数据进行监督微调。双方利用了近百万对高质量法律知识问题数据，

第三步——优化保证输出结果质量和可靠性。

通用型及场景特定型的工程化优化方案，提升法律问答的有法可依和引用法条内容的准确性。

PowerLawGLM 法律专业能力—法律问答

快递被寄丢了怎么办？



如果快递被寄丢了，可以采取以下措施：

1. 及时联系快递公司：发现快递丢失后，第一时间联系快递公司，提供快递单号、收寄方信息、物品名称等相关信息，让客服人员及时进行处理。可以通过快递公司的客服电话、官方网站、微信公众号等渠道进行投诉和查询。
2. 协商解决：联系快递公司后，可以与快递公司协商解决问题，要求全程调监控，如果有一定的机会找回来，也可以要求赔偿。如果找不回来，只能选择赔偿了。
3. 提起诉讼：如果对快递公司的赔偿方案不满意，可以进行申诉或起诉。建议先咨询专业的律师，了解自己的权利和义务，并在法律的范围内进行处理。
4. 注意保价和包装：在寄送重要物品时，建议购买快递保险以获得更好的保障，同时在包装时注意物品的完整性和价值，选择合适的快递公司。

如果快递被寄丢了，要及时采取措施解决问题，保留好相关证据，以便在协商或诉讼时能有据可依。

快递丢了具体处理方式为若是快递员没有派送到指定地点的，应当由快递公司承担责任。在快递服务过程中，快件发生延误、丢失、损毁和内件不符的，经营快递业务的企业应当按照与用户的约定，依法予以赔偿。法律依据：《快递市场管理办法》第二十条

在快递服务过程中，快件(邮件)发生延误、丢失、损毁和内件不符的，经营快递业务的企业应当按照与用户的约定，依法予以赔偿。

企业与用户之间未对赔偿事项进行约定的，对于购买保价的快件(邮件)，应当按照保价金额赔偿。对于未购买保价的快件(邮件)，按照《中华人民共和国邮政法》、《中华人民共和国合同法》等相关法律规定赔偿。

PowerLawGLM 法律专业能力—合同生成

法律合同生成 —— 家具买卖合同

帮我起草一份家具买卖合同，乙方是北京红源家具有限责任公司，购买的家具分别是红源牌实木餐桌和实木椅
议解决方式选择诉讼，管辖法院约定为北京市海淀区人民法院。

家具买卖合同

买方（甲方）：_____

统一社会信用代码：_____

法定代表人：_____

注册地址：_____

授权代表：_____

卖方（乙方）：北京红源家具有限责任公司

统一社会信用代码：_____

法定代表人：_____

注册地址：_____

授权代表：_____

就甲方向乙方采购家具有关事项，双方本着平等互利的原则达成以下合意：

一、产品条款

产品一品牌：红源牌

型号：实木餐桌

单价：800元

数量：50件

小计：_____

产品二品牌：红源牌

型号：实木柜子

单价：950元

数量：50件

小计：_____

二、价款

本合同总金额为￥_____元（大写：人民币_____元整）。合同总金额为乙方提供货物并完成其他相关服务的全部价格（包括但不限于运输费用、保险费用、装卸费用、保管费用、报关费用等）。本合同金额是甲方依据本合同应向乙方支付的全部价款，除本合同金额外，甲方不向乙方支付任何其他费用。

三、货款支付

1.在甲乙双方签订合同后五个工作日内，甲方预付合同总金额_____%的合同款￥_____元（大写：人民币_____元整）；

2.货到甲方指定地点交付并经甲方验收合格后_____个工作日内支付合同总金额_____%的合同款￥_____元（大写：人民币_____元整）；

3.剩余_____%作为质保金，质保金为￥_____元（大写：人民币_____元整），在质保期满后_____个工作日内付清。

四、付款方式

1.甲方应以汇款方式向乙方付款，汇入乙方账户：

户名：_____

地址：_____

纳税人识别号：_____

开户行：_____

银行账号：_____

电话：_____

2.乙方在收款后向甲方开具相应增值税专用发票，甲方开票信息如下：

户名：_____

地址：_____

纳税人识别号：_____

开户行：_____

银行账号：_____

电话：_____

如上述付款方式、账户及开票信息发生改变，买卖双方应提前3个工作日以书面形式告知对方。

五、交货时间及地点

1.交货时间：乙方在签订合同后_____个工作日内按照合同约定交货地点送货。

2.交货地点：_____

3.收货负责人：买方委派_____作为收货负责人，联系电话：_____。

六、交货方式

乙方送货，运费由乙方负担。产品在运输过程中，未曾到达买方指定交货地点前，发生损坏或灭失的风险由乙方承担；产品自运至合同指定地点经甲方验收合格后，发生人为损坏或灭失的风险由买方承担。

七、产品接收和验收

1.乙方依合同所订交货时间将产品运至指定地点时，甲方应及时组织对产品进行验收。

2.买方应对产品的商标、数量及款式等外观特征及有无《家具合格证》进行验收，甲方发现问题应当场提出，并由双方协商达成解决方案。若产品外观质量或数量达不到合同要求，甲方收货负责人有权拒绝接收。

八、质量保证

1.产品应附符合GB5296.6要求的《家具使用说明书》，达到《家具使用说明书》中明示的执行标准，且不低于样品同等质量。

2.对于产品的质量问题，应按照《北京市家具产品修理、更换、退货责任规定》执行，如果乙方作出对甲方更有利的责任承诺，按照该承诺执行。

3.产品质保期为一年，自甲方验收合格之日起算。

4.质保期内乙方应在接到甲方报修通知后24小时内派人上门修理。

九、违约责任

1.乙方非因不可抗力（如地震、火灾、风灾等）逾期送货，每逾期一日，应按照合同产品总额的_____%向甲方支付违约金。逾期送货迟延超过_____日的，甲方有权解除合同。如乙方逾期交货给甲方造成损失的，由乙方承担赔偿责任。

2.经国家认可的家具检测机构检测，家具的有害物质限量不符合国家或北京市有关标准的强制性要求的，甲方有权无条件退货，乙方应按照合同产品总额的_____%向甲方支付违约金，违约金不足以弥补甲方损失的，包括但不限于检测费、交通费、误工费等，乙方还应承担赔偿责任。

3.本合同经买卖双方签字盖章生效后，任意一方因不可抗力单方面提出解除合同，应向对方支付合同总金额_____%的违约金。

十、争议解决方式

本合同项下发生的争议，由双方协商，协商不成的，可依法向北京市海淀区人民法院起诉。

十一、其他

1.本合同经买卖双方盖章（签字）后生效。

2.本合同正本一式二份，双方各执一份，具有同等法律效力。报价单作为本合同附件具有法律效力。

3.合同未尽事宜，由双方签订书面补充协议，补充协议与本合同不一致或相冲突的，以补充协议为准。

4.本合同以及合同履行过程中涉及的补充协议、结算文件、付款计划、欠款说明等一切文件必须由甲方签字并加盖公章方为有效，合同另有约定除外。

| 行业大模型探索

- 1. 通用模型基础知识 + 对话能力**
- 2. 领域知识和数据增强模型专业化能力**
- 3. 结合行业知识库等进一步保证准确性**

| Takeaways

- ChatGLM-6B 是一个开源的、支持中英双语的对话语言模型
- ChatGLM-6B 可以在消费级的显卡上进行本地部署
- 利用领域数据对模型微调能够进一步增强模型专业领域能力

谢谢！



<https://github.com/THUDM/ChatGLM2-6B>
<https://github.com/THUDM/ChatGLM-6B>
<https://github.com/THUDM/VisualGLM-6B>
<https://chatglm.cn/>

