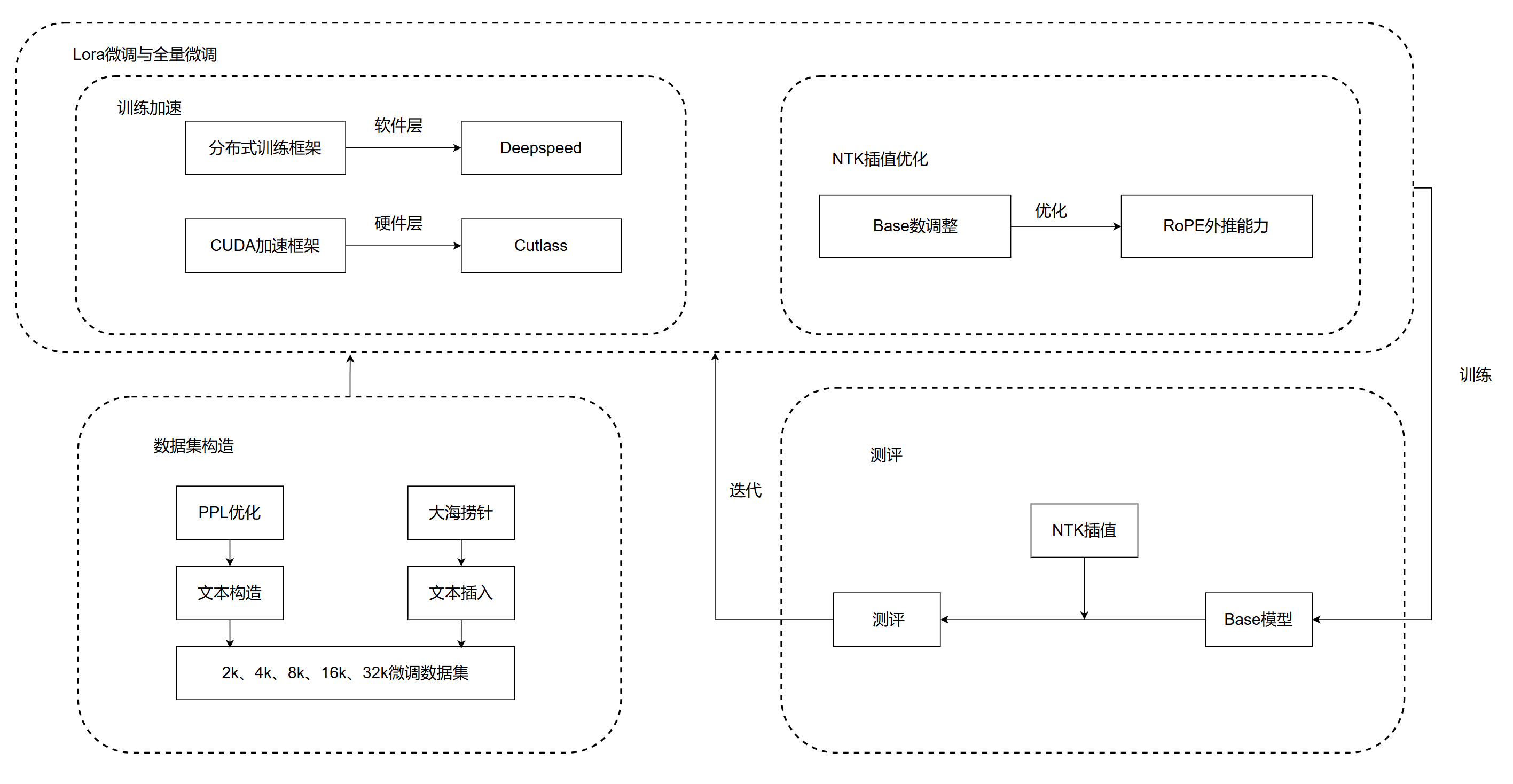
#### 数字金融训练营 技术方案 第5组

#### 1、优化目标：

1. 尽最大可能降低金融场景文本困惑度
2. 针对“大海捞针”测试，尽最大可能优化测试结果惩罚系数

#### 2、方案架构图：



图：总体方案图

项目整体方案分为以下几个模块：

**（1）数据集构造：**针对两个特定优化目标，经过特定分析后我们认为，要优化第一个目标，首先要使用输入-输出相同的数据集来训练模型对长文本的构造能力。针对第二个目标，我们使用文本插入的方式构造数据集，将含正确答案的’针‘插到’缝‘中，构造对应的问答对数据集用于增强模型的发现特定文本能力。考虑长文本的情景，我们的都使用2k、4k、8k、16k、32k文本长度的数据集，且数据集大小为递减，方便后期训练。

**（2）Lora微调与全量微调：**在构造好对应数据集后，采用两种微调方式，分别为Lora微调和全量微调技术对模型进行训练。其中为了优化模型训练速度和效果，我们分别在软件和硬件两个层次进行对应的优化。软件层的优化主要为分布式训练框架deepspeed，硬件层优化为加速框架Cutlass。

**（3）测评迭代优化：**为了优化训练数据的效果，我们并不将数据集全部一次微调，而是多轮按照文本长度输入到模型，并在每次模型优化后得到对应的PPL分数和惩罚系数，根据测评情况继续调整优化。

#### 训练数据集构造：

**（1）PPL任务**

针对PPL优化任务，构建长文本数据集，按照2k，4k，8k，16k，32k的序列长度，各个数据集的序列数量分别为4000（2k），2000(4k)，1000(8k)，500(16k)，200(32k),按照降序选取构造不同序列长度的微调数据集，防止可能的过拟合问题，并对齐各个stage训练的 token量。构造lora-finetune,reconstruct文本任务。

数据集举例，rawdata为数据集提供的plaintext：

rawdata- > {'instruction':rawdata, 'input':' ','output':rawdata}

通过这种方式获取的数据集可以很好的训练大模型对长文本的理解能力。

**（2）大海捞针任务**

针对大海捞针任务，在finetune-ppl任务之后进行继续指令微调构建对应的数据集。例如，在无关文本中插入关键语句构建大海捞针型语句，并生成关键语句的对应问句；优化prompt的设计，融入CoT-prompt.这部分的数据集有两种构建方式。

1. **第一种：**
2. 通过Textrank方式抽取长文本，生成对应的摘要作为目标文本，再根据对应摘要使用LDA抽取关键词，生成随机插入的“针”；
3. 构造长文本-摘要-关键词对，然后打乱文本-摘要对应关系，使得摘要随机插入不同的长文本，而关键词作为答案仍与文本匹配，作为答案训练文本。

举例：

plaintext---->summary---->keyword

'instruction':find the keyword in plaintext

'input':plaintext+summary

'output':keyword

1. **第二种**

第二种为通过人工构造10个优质问答对，将答案插入不同长文本中进行构造。

举例：

'instruction'：问题

"long\_text"+“内容主旨”+"long\_text"

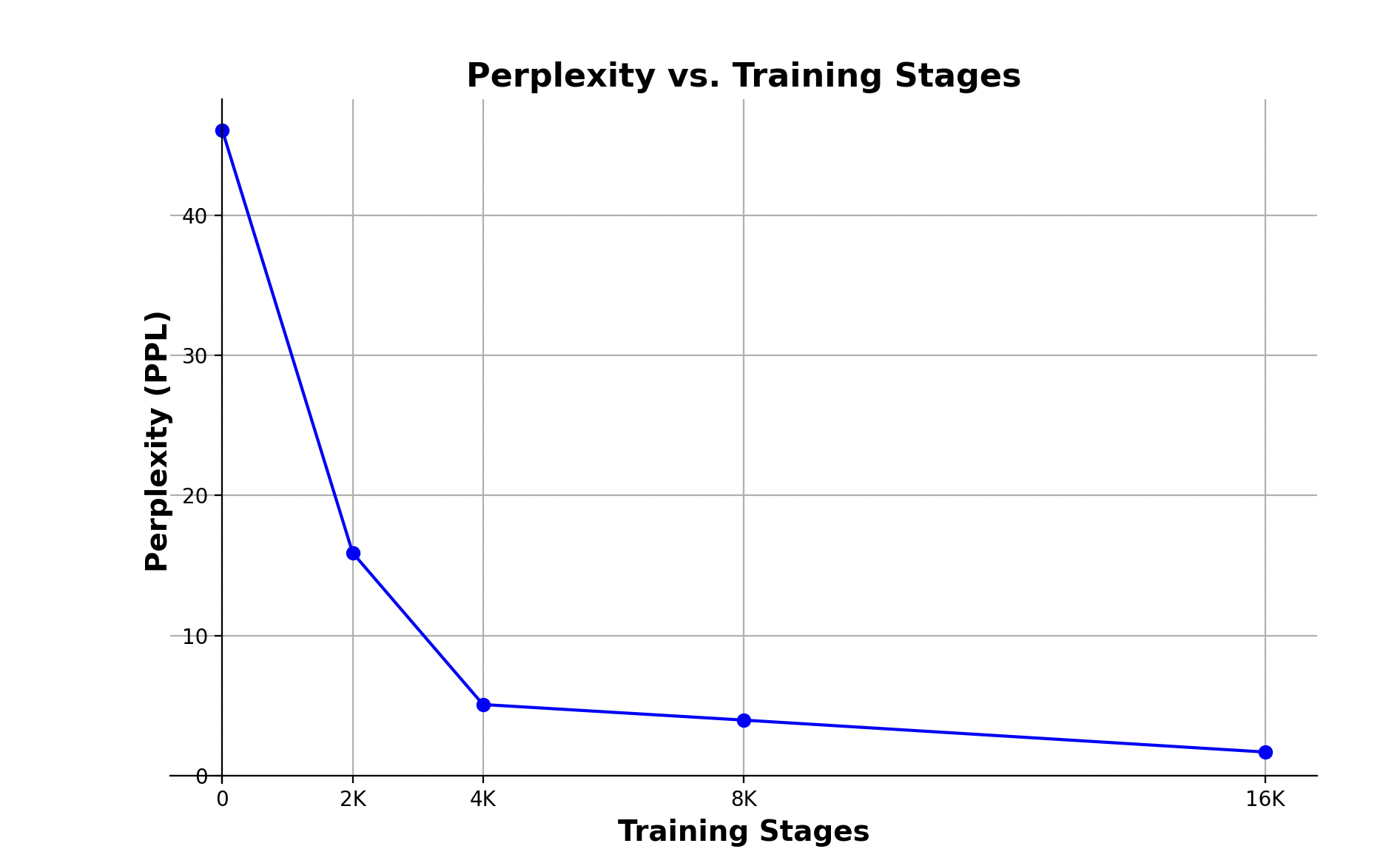
'input':插入随机答案后的长文本

'output': 答案

##### 4、训练过程：

**（1）PPL任务**

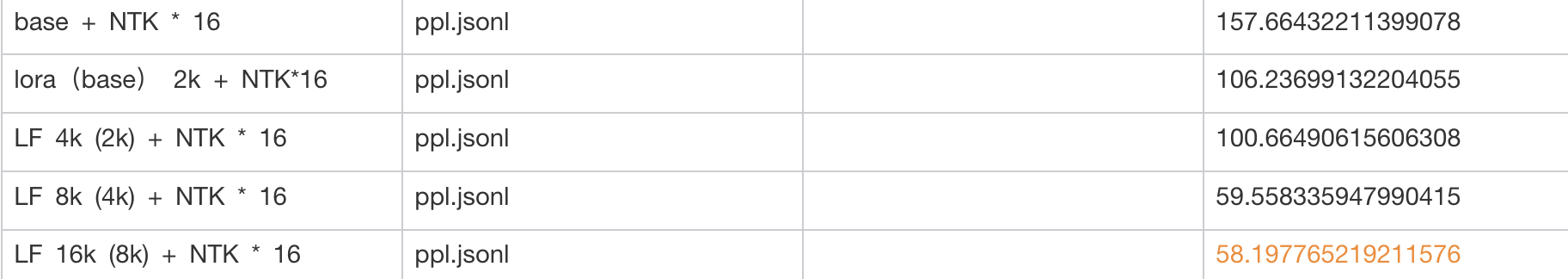
* Stage1：基于原始1.3Bbaseline模型，从2k，4k，8k，16k，32k数据集迭代优化微调baseline模型完成重建任务的训练得到优化后的模型。
* Stage2：基于优化后的模型，利用大海捞针型语句数据集完成最终的训练使得网络学会定位能力和捕获长文本关键句提取能力。
* 2k learning\_rate:1e-5,epoch：2
* 4k learning\_rate:2e-5,epoch：2
* 8k learning\_rate:4e-5,epoch：2
* 16k learning\_rate:8e-5,epoch：2



图：PPL随训练的迭代而降低

**（2）大海捞针任务**

learning\_rate:2e-4,epoch：2



#### 5、PPL任务优化

**（1）NTK插值+Finetune**

模型的长上下文能力不仅体现在文本较长的时候，模型的PPL依然能保持在较低的水平，还体现在对于长上下文输入，模型依然能够进行准确的阅读理解和推理。

比如位置编码相关的线性插值、NTK插值等，虽然使得模型在形式上支持了长上下文的任务，但是在任务的准确性上效果却不佳。

即模型天然更容易关注输入文本的开头和结尾部分的信息，而更容易忽略中间部分的信息，注意力迷失在大量无关内容上。

##### （2）长度外推

位置编码：

Transformer具有置换不变性，乱输入序列的顺序对输出结果不会产生任何影响。因此需要位置编码，即把位置信息引入输入序列中，打破全对称性。

RoPE：

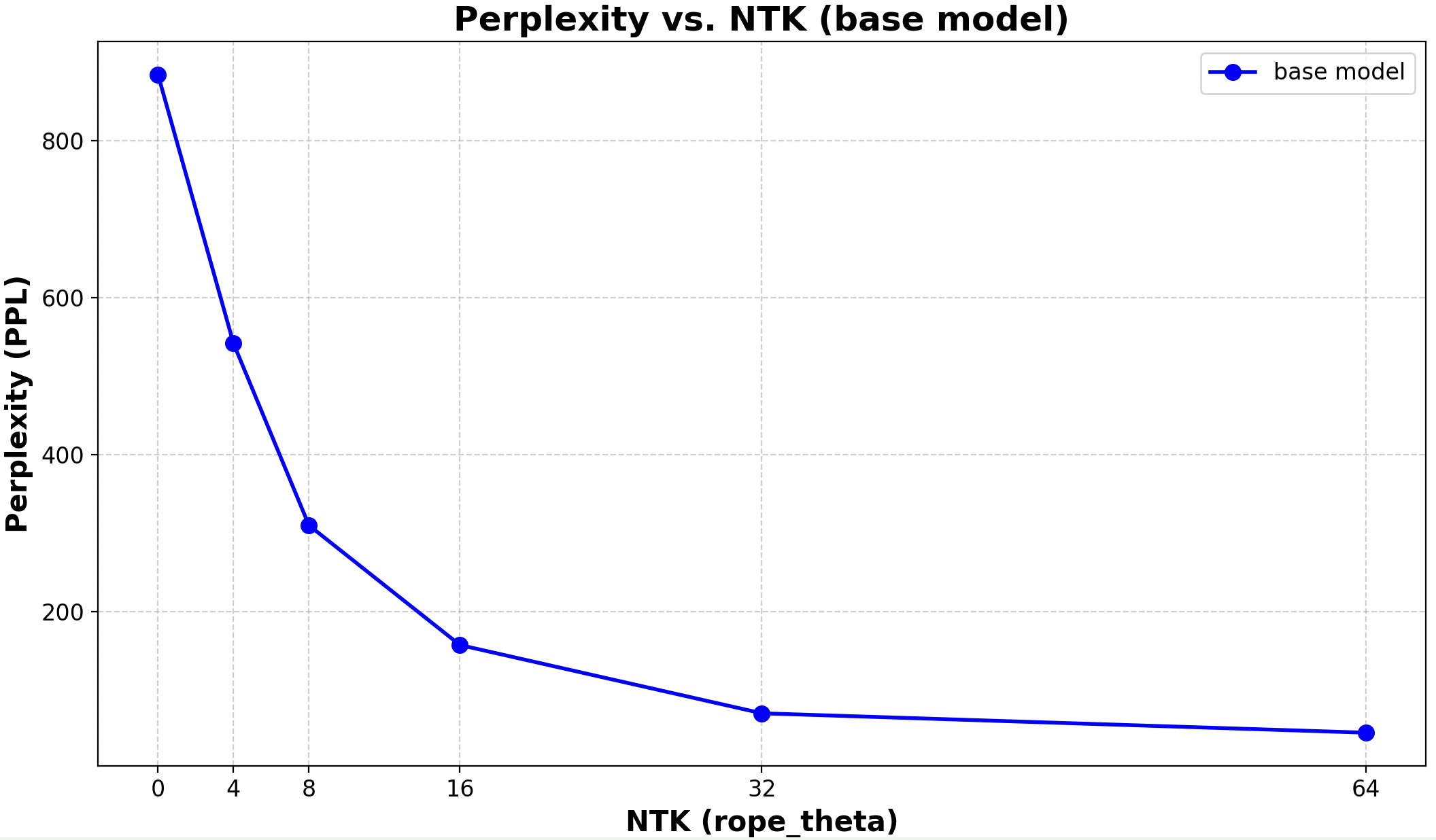
2k -> 4k -> 8k -> 16k长度训练集，先短后长，先易后难，curriculum training。

绝对位置编码，向输入xk引入(concatenate)位置索引k的one hot向量。

旋转式位置编码是指在构造查询矩阵q和键矩阵k时，根据其绝对位置引入旋转矩阵R

我们的位置编码方案：

1. inference阶段：调整不同base大小，在无需微调的情况下，直接降低PPL。



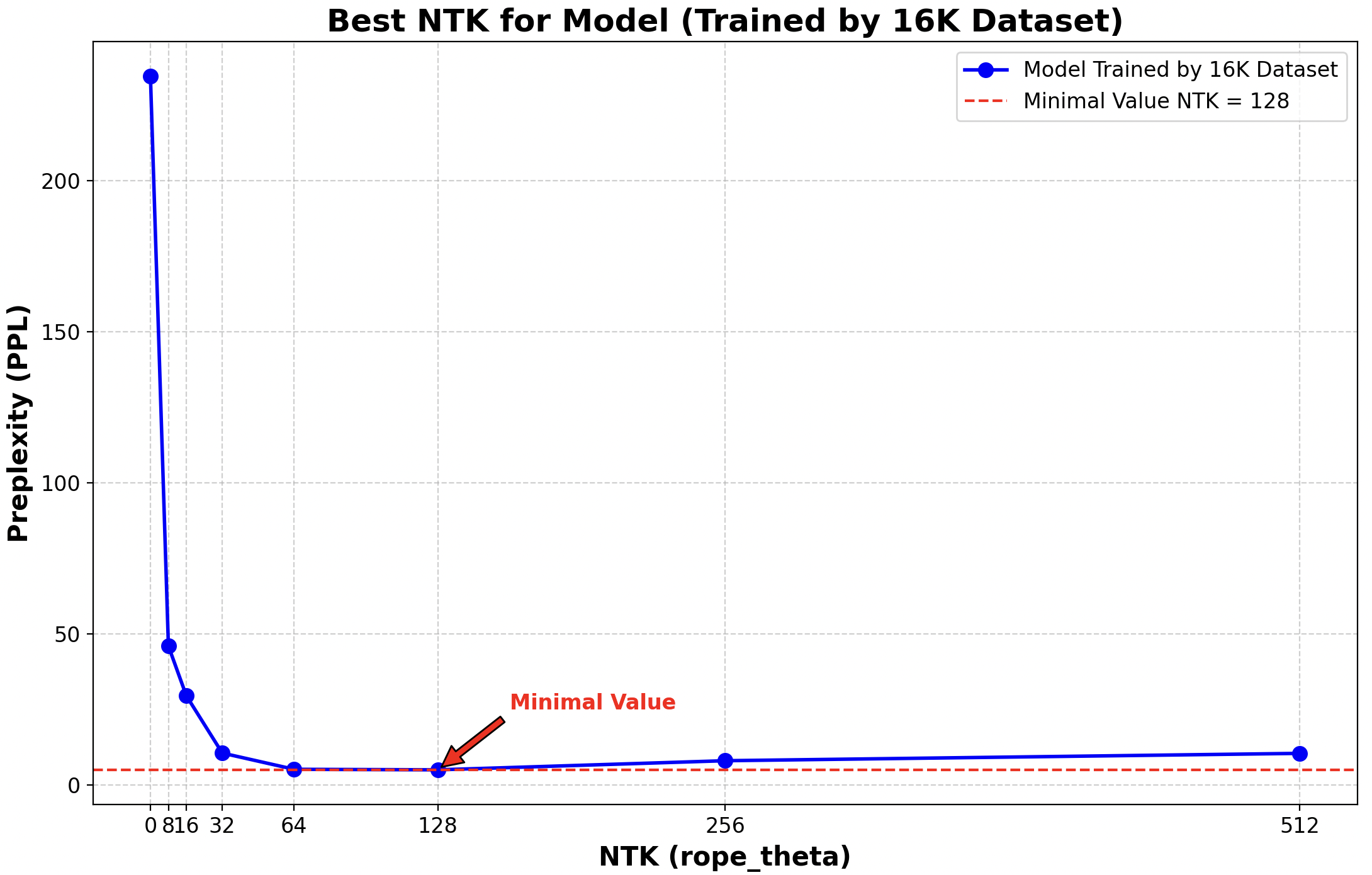
图：inference时RoPE base与PPL的关系

表：inference时RoPE base与PPL的关系

|  |  |  |
| --- | --- | --- |
| 模型 | 测试集 | PPL |
| base | ppl.jsonl（seq len is 163365） | 883.6925171613693 |
| base + NTK \*4 | ppl.jsonl | 541.581274497509 |
| base + NTK \*8 | ppl.jsonl | 309.96350114345546 |
| base + NTK \* 16 | ppl.jsonl | 157.66432211399078 |
| base + NTK \* 32 | ppl.jsonl | 70.73547735214233 |
| base + NTK \* 64 | ppl.jsonl | 46.04955933094024 |

表：inference时三种改RoPE的方法

|  |  |  |
| --- | --- | --- |
| 模型 | 测试集 | PPL |
| base + NTK \* 16 | ppl.jsonl | 157.66432211399078 |
| base + linearNTK \* 16 | ppl.jsonl | 294.701322555542 |
| base + dynamicNTK \*16 | ppl.jsonl |  |



图：16K数据集训练后的模型，NTK=128时PPL取最小值

#### 该图为16k长度数据集下的实验结果，从图中可以看出，当NTK为128时，PPL值取得最小值。

#### 6、大海捞针任务优化

在短窗口(2k)上进行stage1-finetune，根据An S, Ma Z, Lin Z, et al. Make Your LLM Fully Utilize the Context[J]. arXiv preprint arXiv:2404.16811, 2024.进行5000序列长度的CoT- prompt-instruction微调，在stage2- finetune，直接加入大数据文本（50000序列微调），提升模型实际的远距离建模能力，对于实际“针”进行提取

领域之间有竞争关系，最好和原预训练模型所用的分布保持一致。仔细考虑数据长度和领域分布的影响。通常来说，某些领域天然会有更高比例的长文本数据，比如书籍、论文和github，而一些其他领域的长数据就较少，比如新闻。如果直接从整体数据中挑选长数据而忽略领域分布，就可能造成训练数据在领域分布上的偏移。



#### descript

图：指令微调loss曲线图

#### 7、过程中遇到的问题以及解决措施

1. 初始并行化框架在训练大于4k文本时，出现显存卡死无法继续训练

问题分析：使用的框架不支持4k以上文本加载，换用llama factory框架加载

1. 解决大海捞针问题中输出有带prompt导致问题

问题分析：测试的prompt问题存在自带模板，导致生成文本自带prompt

#### 8、未来展望：

1. 与RAG技术结合更好的解决大海捞针问题
2. 分布式训练的优化以及调优问题