2025年Dian团队春招算法方向考核试题

> ❕ **注意事项**
>
> - 编程前请注意对算法原理的理解。
> - 编码过程中请注意对代码细节的掌握。
> - 编程完成后请按题目条理上传至个人github仓库并在仓库提交后设置为公有状态方便我们查看；体量较大的模型文件可上传至Huggingface个人模型仓库，在最后需要提交你的github与Huggingface仓库链接。
> - 部分题目涉及到模型训练，T4 16GB显存的训练卡即可，如果你没有可用的GPU，可通过以下途径获取：
>   - 免费获取，有额度限制，但基本能满足需求：
>   - Google Colab:https://blog.csdn.net/ychpython/article/details/127293104
>   - 阿里天池：https://blog.csdn.net/m0_75079597/article/details/138856414
>   - 在AutoDL等算力租赁平台按量购买，约2RMB/时。

一、随机森林的理解与实现

即使是在深度学习大行其道的今天，经典机器学习仍是人工智能的重要组成部分。随机森林是集成学习的一种方法，通过构建多个决策树并将其结果进行集成，即可提高模型的性能和稳定性。通过自主实现一个随机森林，可以对决策树与随机森林有一个较为深刻的理解。

在本题中，你需要使用numpy从头实现一个随机森林算法，再用其拟合一个简单的数据集，接着对提供的特征的重要性进行评估并可视化。

1-1 逐步完成随机森林模型的构建

- 决策树的构建：请注意思考任务特性，从ID3/C4.5/CART决策树中选取你认为最合适的一个，构建时需要考虑特征选择、节点分裂准则、停止条件等因素。
- 随机森林的构建：需要考虑集成学习随机性的引入，包括Bootstrap采样、特征子集选择等。
- 模型预测方法的实现。

1-2 模型训练与模型评估

- Iris(鸢尾花数据集)是很经典的分类数据集，包含4个数值型特征，3个目标分类，共有150条均衡分布的记录。你需要自行划分数据集，用上述自行构建的随机森林拟合训练集，并自行实现合适的评价函数，对你所实现的随机森林在Iris数据集上的拟合效果进行评估。
- 请从如下链接下载数据集：https://archive.ics.uci.edu/dataset/53/iris。

1-3 特征重要性评估

在拟合数据集后，请你使用一种方法给出各特征对于预测任务的重要性，并对评估结果进行可视化。

二、Bangumi评论分数预测器的训练

经过了上一题从随机森林的构建、训练与评估的端到端的流程，相信你已经对机器学习的一般范式有了较为深刻的理解，甚至可能觉得数据量太小、拟合太容易。接下来，你将通过训练一个深度学习模型走进深度学习世界。

Bangumi 是一个专注于动漫、游戏、音乐、小说等文化内容的社交网站（https://bangumi.tv/）。用户可以在平台上记录、评分、评论和讨论各种ACG作品，并查看其他用户的评价和推荐。BERT是一个非常经典的具有通用语义理解能力的预训练模型。在此任务中，你需要完成数据爬取、数据清洗、模型训练、评估与优化的流程，得到一个评论分数预测器，针对输入的某条评论对其评分进行预测。

2-1 数据爬取与清洗

- 在实际工作中，数据通常不是现成的，因此需要你自行获取。不过，我们提供了爬虫脚本以供数据爬取，你可对该脚本作定制化调整。请注意限制最大请求频率，避免被服务器拦截。
- 数据质量对模型训练效果影响深远，请你完成对爬取数据的清洗。
- 爬虫脚本文件获取链接：https://github.com/rica451/catch。

2-2 模型微调

- 在获取语料数据集后，你需要对预训练的中文BERT模型（模型ID为google-bert/bert-base-chinese）进行微调训练，请你使用pytorch编写训练脚本。
- BERT模型获取链接：https://huggingface.co/google-bert/bert-base-chinese，如有访问问题，建议使用镜像站，请参考 https://blog.csdn.net/weixin_40959890/article/details/140319652。

2-3 模型评估与优化

- 我们会提供一个包含text与label属性的验证集，以便对微调后的模型效果进行验证。你可针对模型在验证集上的表现进行优化，优化方向包括数据质量、数据规模、模型训练细节等。
- 验证集可从如下仓库获取：https://github.com/rica451/catch。

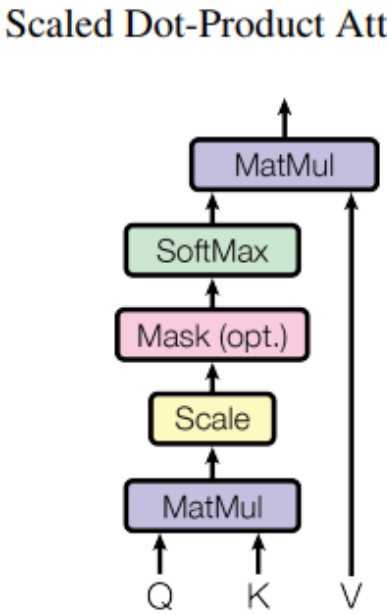
- 为了对你所训练的模型进行测试，你需要将训练后的模型上传至huggingface平台，我们会在最后拉取你的模型并在内部测试集上进行推理与评估。

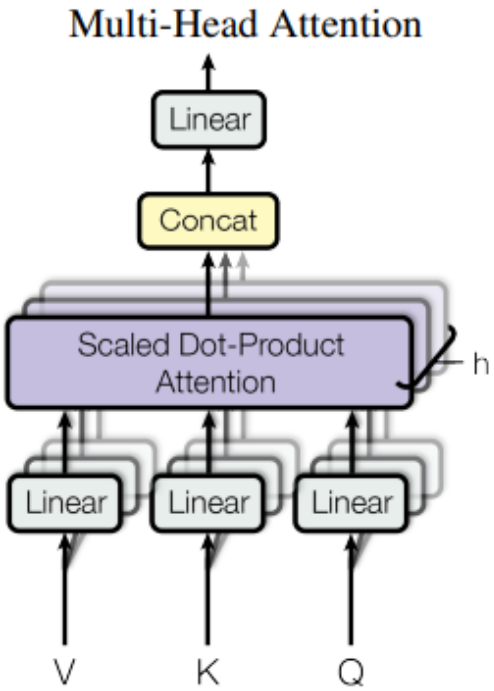
三、注意力机制及其变体的理解与实现

BERT实际上是Transformer中的Encoder部分，而Transformer中的一个重要机制是注意力机制(Attention)，它通过将多个隐藏层状态映射并执行权重分配，保证了采样域中的有效训练。

3-1 多头注意力机制的实现

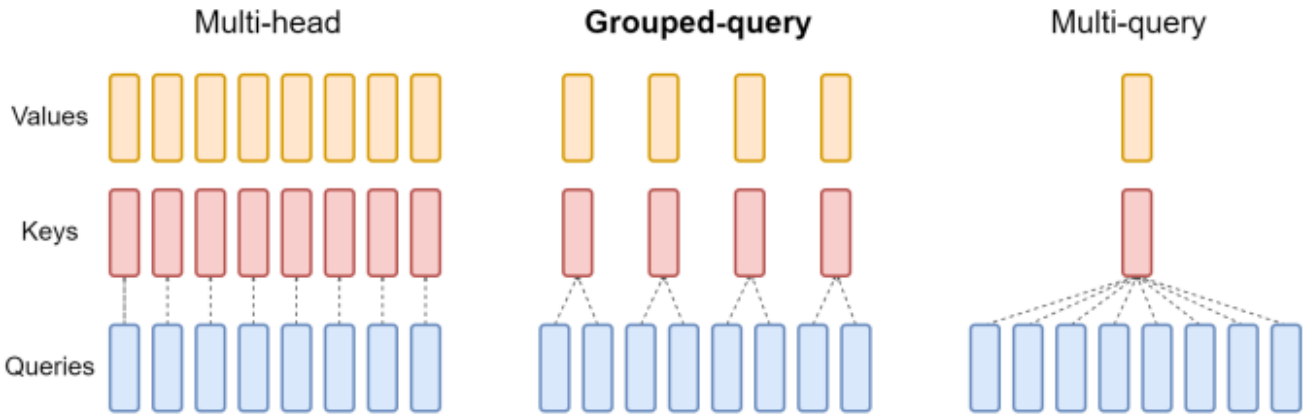
请你使用pytorch实现基本的多头注意力机制，并计算一个随机矩阵的注意力权重。此过程中请注意对KV Cache的理解。





3-2 注意力机制变体的实现

GQA(Group Query Attention)与MQA(Muiti-Query Attention)是多头注意力机制的改进版本，主要目的是为了减小Attention在推理过程中KV Cache的空间占用。请你使用pytorch实现GQA与MQA，并计算与3-1相同的随机矩阵的注意力权重并注意对比。



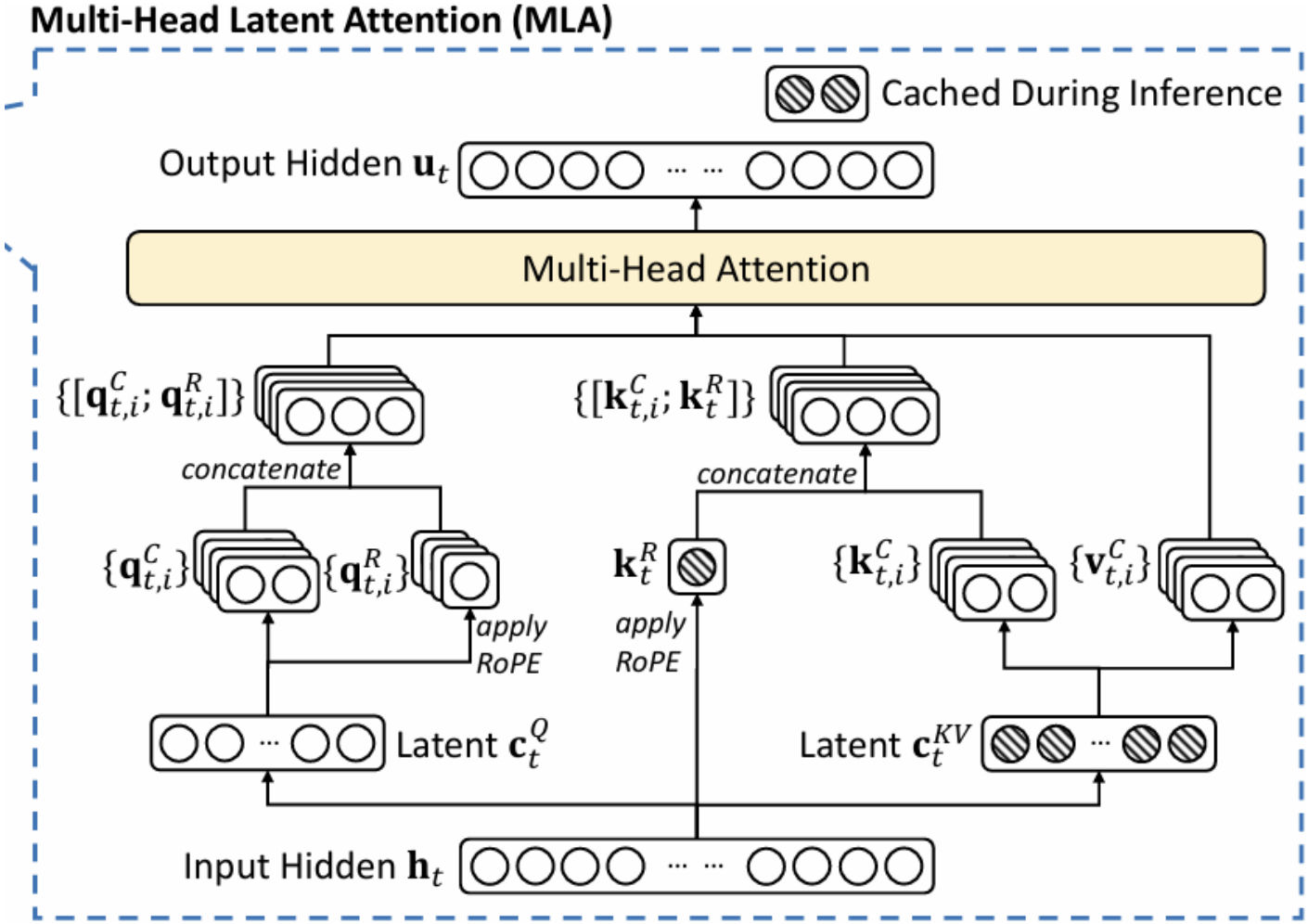
附加题：多头潜在注意力机制的理解与实现

相信你一定听说过最近特别火热的DeepSeek，DeepSeek系列模型所提出的一个关键技术是**多头潜在注意力机制**(Multi-Head Latent Attention, MLA)。“MLA 被视为 GQA 的一般化，它用投影矩阵的方式替代了 GQA 的分割、重复，并引入了一个恒等变换技巧来可以进一步压缩 KV Cache，同时采用了一种混合方法来兼容 RoPE。”

- 请你理解多头潜在注意力机制的原理。
- 在理解的基础上，尝试使用pytorch实现MLA。（选做）

阅读资料：

- DeepSeekV2技术报告
- 缓存与效果的极限拉扯：从MHA、MQA、GQA到MLA - 科学空间|Scientific Spaces


Multi-Head Latent Attention (MLA)
Cached During Inference
Output Hidden $\mathbf{u}_t$
Multi-Head Attention
$\{[\mathbf{q}_{t,i}^C; \mathbf{q}_{t,i}^R]\}$
$\{[\mathbf{k}_{t,i}^C; \mathbf{k}_t^R]\}$
concatenate
concatenate
$\{\mathbf{q}_{t,i}^C\}$
$\{\mathbf{q}_{t,i}^R\}$
apply RoPE
$\mathbf{k}_t^R$
apply RoPE
$\{\mathbf{k}_{t,i}^C\}$
$\{\mathbf{v}_{t,i}^C\}$
Latent $\mathbf{c}_t^Q$
Latent $\mathbf{c}_t^{KV}$
Input Hidden $\mathbf{h}_t$