Notes on LLM

LaTeX by Junfan Zhu

本文是基于浙大毛玉仁老师《大模型原理与技术》课的学习笔记。[1]

2025 年 2 月 24 日

目录

1 语言模型 **2**
1.1 语言模型 . . . 2
1.2 RNN 与 Transformer . . . 2
1.3 采样与评测 . . . 2

2 LLM 架构 **3**
2.1 基于 Transformer 的三种架构 . . . 3
2.1.1 只用 Transformer 的 Encoder (BERT, RoBERTa) . . . 3
2.1.2 用 Transformer 的 Encoder-Decoder (T5, BART) . . . 3
2.1.3 只用 Transformer 的 Decoder (GPT, LLaMA) . . . 3
2.2 非 Transformer 架构：Mamba . . . 4

3 Prompt 工程、思维链 **4**

4 参数高效微调 Parameter-Efficient Fine-Tuning (PEFT) **4**
4.1 三种 PEFT . . . 4
4.1.1 参数附加 . . . 4
4.1.2 参数选择 . . . 5
4.1.3 LoRA 低秩适配 Low-Rank Adaptation . . . 5
4.2 PEFT 应用 . . . 6

5 模型编辑（三体的思想钢印） **6**
5.1 外部拓展 . . . 6
5.2 内部修改 . . . 6
5.3 应用 . . . 7

6 检索增强生成 Retrieval-Augmented Generation (RAG) **7**
6.1 4 种 RAG 架构 . . . 7
6.2 知识检索 . . . 7
6.3 生成增强 . . . 8
6.4 降本增效 . . . 8

1 语言模型

1.1 语言模型

1. 基于统计的语言模型。N-grams 可以最大概率生成当前语料库，n 是拟合语料库的能力与未知文本泛化能力的权衡，n <= 5。观测长度有限、泛化能力不足。

2. 基于学习的语言模型

   (a) 归纳偏置 Inductive Bias：限制对某些假设进行选择，如上下文间存在关联。SVM：超平面间距，CNN：局部感受野，RNN：时间序列，Transformer：稀疏注意力

   (b) 损失函数：交叉熵损失用连续函数模拟 0-1 损失，为了减少损失用 1/0 阶优化（梯度下降）学习算法。

   (c) 泛化误差：机器学习目的是减小泛化误差（未知损失的期望），即真实误差。概率近似正确 Probably Approximately Correct (PAC)：当样本数量一定时，机器学习模型以一定概率达到近似正确。

1.2 RNN 与 Transformer

1. RNN 循环神经网络（包含环路）。串行输入时，前面的元素被循环编码成隐状态，并叠加到当前的输入上。有记忆历史的压缩器，是时间维度上各种嵌套的复合函数，对历史信息的考虑是螺旋式前进的。

2. LSTM。由于大量矩阵联乘，训练越长越容易梯度爆炸/衰减，为了解决这个问题，提出带有门控机制的 LSTM，相当于把 RNN 套娃的复合函数乘法变成状态累加，用 GRU[遗忘门（往事）+ 输入门（选择性聆听新闻）] 进行合理状态累加，输出门实现整合。

3. Transformer，基于 Attention 的模块化构建的神经网络（搭积木），支持并行输入所有文本。两个模块：

   (a) 注意力模块（对上下文通盘考虑）= 注意力层（加权平均地把前文信息叠加到当前状态，权重是学出来的）+ 残差连接（防止梯度消失）+ 层正则化（可以加速网络训练，提高泛化性能）。

   (b) 全连接前馈模块（占 2/3 参数，掌管记忆）。

4. 自回归过程：将第一个词输入 RNN/Transformer，预测下一个词，不断迭代地输出一段话。有错误级联放大、串行效率低的问题，用 Teacher Forcing（防止学错）归纳偏置解决。但 Teacher Forcing 导致曝光偏差 Exposure Bias（训练过程与推理过程的差异），导致幻觉。

1.3 采样与评测

1. 采样。根据预测概率值从词表中选择输出的词元，用 Temperature 在不同场景中调节随机性。

   (a) 概率最大化（生成常见文本和废话文学）

   (b) 随机采样（Top-K 要么胡言乱语要么枯燥无味；Top-P 设定阈值来选取，可以减少胡言乱语和枯燥无味）。

2. 评测

(a) 内在评测。指标困惑度 Perplexity，困惑度减小则胡言乱语减少。

(b) 外在评测。基于统计指标 BLEU (Bilingual Evaluation Uncertainty)，计算多层次 n-gram 匹配精度的几何平均；基于上下文嵌入 Contextual Embeddings 评测 BERTScore（用 Precision, Recall, F1 评测）；基于大模型 Prompt Engineering 评测。

2 LLM 架构

2.1 基于 Transformer 的三种架构

2.1.1 只用 Transformer 的 Encoder (BERT, RoBERTa)

= 输入编码 + 特征编码 + 任务处理。完全双向注意力可以捕捉每个 Token 上下文，小模型专注于判别任务，不适合生成任务。

1. BERT 结构与 Transformer encoder 一致，但规模更大，预训练用掩码语言模型 Masked Language Model（完形填空）和下文预测 Next Sentence Prediction（两句话是否连续）学习生成上下文嵌入。

2. RoBERTa 比 BERT 更大，移除了 BERT 的下文预测，将静态掩码强化为动态掩码语言建模，增加训练多样性。

3. ALBERT 是缩小版 BERT，用 LoRA 思想（对 Embedding 模块进行参数因子分解，对 Attention, FFN 模块跨层参数分享参数复用，减少参数量），把上下文预测替换为句序预测。

2.1.2 用 Transformer 的 Encoder-Decoder (T5, BART)

用交叉注意力机制实现两个自注意力模块（Encoder-Decoder）交互 =（编码器内部的 & 解码器对编码器的）完全注意力 +（解码器内部的）下三角注意力。模型太复杂故不利于 scaling law。

1. BART (Bidirectional AutoRegressive Transformers) 用多样化任务（遮挡删除打乱旋转）来理解语义，然后将预训练内功迁移到不同场景。

2. T5 (Text-to-Text Transfer Transformer) 用多任务有监督微调（构造合理的输入前缀）让任务大一统（用 promt 工程适配多种下游任务），用 Span 级别掩码（完形填空有意义的语义单元的几个 token 掩码）。

2.1.3 只用 Transformer 的 Decoder (GPT, LLaMA)

= 输入编码 + 特征编码 + 输出生成，但省略了每个解码模块中的交叉注意力子模块（解码对下文是掩盖的，下三角注意力，attention 只能注意到上文不能注意到下文）。适合生成任务。

1. GPT。GPT-1 是带掩码的单向自注意力机制，GPT-2 提升了预训练数据的数量质量（可以不微调直接推理），GPT-3 显著增加模型规模提升任务泛化能力涌现出上下文学习 In-Context Learning，InstructGPT 采用 Reinforcement Learning from Human Feedback (RLHF)。

2. LLaMA，Chinchilla Scaling Law 小模型大数据。LLaMA-1 相比 GPT-1，在词嵌入模块用旋转位置编码替代绝对位置编码，在注意力模块用 Pre-Norm 策略对输入正则化，在全连接前馈模块用更光滑的 SwiGLU 替换 RELU 作为 Transformer 激活函数。LLaMA-2 引入 RLHF 并拒绝采样，还用分组查询注意力 Grouped Query Attention 提升计算效率。LLaMA-3 大模型。

2.2 非 Transformer 架构：Mamba

基于选择状态空间模型 Selective State Space Model（非 Transformer）架构：Mamba（RNN 与 Transformer 的折中，与 LSTM 似曾相识）

Transformer 输入窗口长度有限，模型规模随输入序列长度平方次增长，计算成本高。Mamba 在状态空间模型 State Space Model（把 n 阶系统用 n 个 1 阶系统来矩阵表达，把连续的自然语言用 zero-order hold 离散化）基础上加入选择机制，高效处理长序列，又有线性复杂度又没有平方次计算量也没有梯度爆炸。

3 Prompt 工程、思维链

1. 上下文学习 In-Context Learning。直接检索，聚类检索（相似性），迭代检索（前两者 tradeoff，兼顾多样性相似性，但慢）。

2. 思维链。Flat-Scaling curve 现象，对 System-2 逻辑推理任务有瓶颈，因此需要给中间步骤。

   (a) 按部就班（适合计算，不适合反复选择和回溯）：CoT 手工构造解题步骤。Zero-Shot CoT 让大模型一步步思考。Auto CoT 用 Sentence-BERT 对问题库样本进行表征，用 K-Means 筛选例题样本，用 Zero-Shot CoT 生成思维链。

   (b) 三思后行（可以回溯和重新选择）：Tree of Thought (ToT) 拆解衍生评估搜索, Graph of Thought

   (c) 集思广益（汇集多种观点）：Self-consistency（用多种推理路径生成并汇总答案，深思熟虑并对结果投票）, Universal Self-consistency（开放性问题如文本摘要没有统一答案，用 LLM 选最合适答案）

   (d) GPT-o1 不仅在推理端，还在训练时也用 CoT+ 大规模强化学习。谷歌 SCoRe 两阶段学习法，训练模型自我反思、纠正（多轮强化学习）和长推理。用奖励模型指导大规模搜索。

3. Evol-Instruct 用 Prompt 工程构造 prompt（深度、广度、淘汰演化）。大模型驱动的具身智能 Alter3。多智能体斯坦福小镇。

4 参数高效微调 Parameter-Efficient Fine-Tuning (PEFT)

预训练模型难以适配下游任务，指令微调 Instruction Tuning 构建指令数据、监督微调。为解决全量微调的问题，参数高效微调避免更新全部参数，减少更新的参数数量和计算开销。LoRA 甚至可能超越全量微调基线，因为避免过拟合。

4.1 三种 PEFT

4.1.1 参数附加

会改变模型结构，而不像 LoRA 是体外插件。

1. 加在输入 Embedding 中（Prompt-tuning），灵活：内存效率高，多任务能力，缩放特性。额外参数是软提示 Soft Prompt（可在训练中动态调整、可训练、连续的嵌入），Prompt-tuning 引入软提示作为模型输入的一部分，在微调中仅软提示的参数被更新。

2. 加在模型隐藏层，泛化好：参数效率高，任务适应性强，保持预训练知识。

   (a) Prefix-tuning 加在输入嵌入和 Transformer 注意力模块, 衔接到已有参数上，积木雕花

   (b) Adapter-tuning，沿用 Transformer 搭积木，把附加参数也当做灵活的积木（瓶颈 Bottleneck 结构全连接模块 = 上投影矩阵 + 非线性映射 + 下投影矩阵，两个矩阵可压缩特征）加入 Transformer，不变动原有 Transformer 模块。

3. 加在解码时修改模型输出分布，更小代价：只要微调小模型，让小模型访问大模型的输出预测分布，来实现大模型的定制化调整，能四两拨千斤解决大规模模型微调和黑盒模型微调的问题。Proxy-tuning 用两个小模型（专家模型 + 反专家模型），对反专家模型进行微调，在每个自回归步中计算两个小模型的 logits 分布差异，加入代理模型下一个预测词的 logits 分布中，得到微调后的专家模型。

4.1.2 参数选择

1. 基于规则 (BitFit)：微调网络每层的偏置项 Biases 和任务特定的分类头

2. 基于学习 (Child-tuning): 让大模型自己学可训练的参数子集。用梯度掩码矩阵对选中的子网络梯度更新，屏蔽子网络以外的梯度，从而高效微调选择的参数。

4.1.3 LoRA 低秩适配 Low-Rank Adaptation

用低秩矩阵近似原始权重更新矩阵，仅微调低秩矩阵，优势是参数效率、插件化特性（模型结构不变）、跨任务泛化（不同任务插件之间可以互相组合）。

1. LoRA 把 $d*k$ 矩阵分解为 $r*k, d*r(r \ll \min(d,k))$ 矩阵，冻结原模型参数，仅微调两个小矩阵（上投影，下投影）来近似原始矩阵。相当于体外插件（与大模型解耦，可参数隔离），可加在 Attention 层或 FFN 层的投影矩阵上。

2. 性能取决于权重初始化（上投影矩阵 B 用 0 初始化，下投影矩阵 A 用高斯分布初始化）、秩 r（简单任务用低秩，复杂任务用高秩）、施加位置。性能优化主要是梯度内存和优化器内存。

3. 性能改进 (AdaLoRA)：Transformer 不同模块和层中的权重矩阵重要性不同，因此将参数更新矩阵参数化为奇异值分解 SVD 的形式，再通过奇异值剪枝动态调整不同模块的 LoRA 的秩，秩越大代表模块参数越重要，把对秩的控制变成对特征值的控制。

4. 任务泛化 (LoRAHub): 把很多 LoRA 当做原材料存入仓库，使用时选择原材料的比例加权平均组合，根据新任务学习权重，适应不同任务的 LoRA。

5. 训练改进 (QLoRA): 先量化 Quantization 到 4-bit 低精度表示来减小训练成本，再反量化 Dequantization，可以比 LoRA 瘦身 60% 内存占用。

6. 推理改进 (S-LoRA): 可以单卡推上千个 LoRA。因为多个 LoRA 插件共同使用时，一次只能调用一个插件，卸载再添加是串行 I/O 的，很慢。S-LoRA 把输入与 LLM 和 LoRA 参数的运算拆开，每个插件并行计算自己的输出。把 LoRA 插件放入 KVCache，进一步提升硬件性能。把预训练模型 all-reduce 通信与 LoRA 的 all-gather 通信融合，降低通信开销。只适合 Attention 的 LoRA 不适合 FFN 的 LoRA。

4.2 PEFT 应用

1. ReLoRA 通过多次重启 LoRA 可以达到与全量预训练相近的性能。
2. 连续学习也不遗忘，通过插件化（知识小本本）、任务特定、参数隔离。
3. 医疗模型 LoRA 微调，回复医学问题。金融模型 FinGPT 用金融指令数据集 fingpt-sentiment-train 对 LlaMa2-7b 来 LoRA 微调，提供金融预测、个性化建议。法律 ChatLaw.

5 模型编辑（三体的思想钢印）

5.1 外部拓展

将新知存储在外部知识库，与原始模型一起回答。

1. 附加参数 (T-Patcher): 像 PEFT adapter 装备，在最后一层 FFN 添加补丁参数（FFN 可以当做键值存储体，所以最后一层可以添加补丁精确控制其激活。
2. 补丁就像小修改器，只会被相关输入激活），是 1 对 1 的（不同于 CALINET 是 1 对多的，每次训练 100 个知识点）每个补丁独立训练，适合连续编辑。
3. T-Patcher 为每个要编辑的 Token 都添加补丁，从准确性（确保补丁被激活，确保补丁编辑正确）、局部性（确保无关输入不激活补丁、确保相关无关输入具有激活值差距）角度设计损失函数。

5.2 内部修改

1. 元学习
   (a) KnowledgeEditor 用双层优化（外层基于具体任务更新元知识，内层固定元知识），将超网络 Hypernetwork（指导模型如何调参）作为元知识，学习如何更新模型参数来修改特定知识，从而保证局部性 + 准确性。
   (b) MEND 通过低秩分解，用更小的超网络。
2. 定位编辑: 先定位到要编辑的模块，然后对这块神经元修改。
   (a) KnowledgeNeuron 把全连接层每个中间激活值作为知识神经元，修改它的键向量来编辑。通过计算神经元在预测正确答案的梯度变化，确定哪些神经元在知识表达中起关键作用。
   (b) ROME
      i. 通过因果跟踪实验（正常推理、干扰推理添加噪声、恢复推理即因果效应）来定位与编辑知识最相关的全连接前馈层，求解带约束的最小二乘问题，得到下投影矩阵变化量，实现编辑；通过阻断实验发现 Last Subject Token（e.g. 输入到斑马的马这个字）在中间层的因果效应来自 FNN（而非注意力层，Attention 头只是转移信息复制粘贴的作用，不会阻断知识流动，Transformer 中间层也能互换）。
      ii. 知识存储假设：早起层收集主体信息并汇聚到 Last Subject Token 的向量表示中，中间层对向量查询并将相关信息融入残差流，末尾层捕获并整理隐藏状态中的信息并输出。根据这些假设，确定键向量（Last Subject Token 在 FNN 激活后的向量）、优化值向量（用损失函数确保准确性 = 交叉熵、局部性 = 编辑前后模型输出 KL 散度）、插入知识（最小二乘问题，确保编辑正确且对以前问题影响不大）。

(c) MEMIT 把 ROME 扩展到大量知识编辑，一次关注多个 FFN 层，批量编辑、效率高。

5.3 应用

1. 知识蒸馏：训练两个辅助模型来模拟遗忘数据（用梯度上升 = 损失函数取反）和保留数据的分布差异，优化原始模型和辅助模型的知识差异。

2. 参数高效模块 PEM (Parameter Efficient Module) 操作否定运算符来破坏内部知识，用专家 PEM 和反专家 PEM 遗忘有害能力。

6 检索增强生成 Retrieval-Augmented Generation (RAG)

应用：多模态（医疗）、Agent（RAG 辅助记忆计划行动）、AI 产品开发（解决准确性、安全性、幻觉的痛点）、知识集成（RAG 模块化 + 微服务架构 = 信息检索微服务）、大数据信息检索（高级索引技术 + 向量数据库）、个性化量身定制（用户偏好、历史、上下文）、系统响应（RAG 把 LLM 和知识库分开，知识库实时更新）、保护隐私（敏感数据仅本地检索）、多种数据（文本图像音频）。

6.1 4 种 RAG 架构

可降低 perplexity，按生成器参数是否可更新，分为黑盒和白盒架构。

1. 黑盒架构，成本低，按检索器是否可微调，分为无微调（In-Context RALM，大模型与检索器完全独立）和检索器微调（REPLUG 让检索器迎合大模型，用语言模型困惑度作为监督训练检索器）。

2. 白盒架构，成本高，按检索器是否可微调，分为仅微调大模型（RETRO，把检索器的知识交叉编码融入大模型，让大模型懂检索器）和协同微调（ATLAS，同时微调大模型和检索器，用 KL 散度损失函数训练二者深度合作，让文档相关性与模型贡献的分布一致）。

6.2 知识检索

1. 判别式检索器，可用向量数据库（有索引）提高效率。

   (a) 稀疏检索器：TF-IDF 词频统计

   (b) 双编码检索器：用 BERT 做问题和文档的两个编码器并计算相似度，但两个编码器无交互、自说自话。

   (c) 交叉编码检索器（效率低）：ColBERT 让两个编码器有交互，用查询和文档间的 Token 级相似度进行对比学习，对双编码器微调。把问题 + 文档一起输给编码器，再外接分类器来打分，这样问题和文档强交互。

2. 生成式检索器，一步到位端到端，把知识库记住，问问题回答文档 ID。

3. 图检索器，更好理解知识间关系。

4. 检索结果重排精选（交叉编码器 Sentence-Transformer）。

6.3 生成增强

1. 是否增强：对输入问题，用训练好的探针（线性分类器）预测问题是已知还是未知。
2. 何处增强：输入端（用 prompt 喂知识，主流方法但费钱）、中间层（用交叉注意力把知识编码到模型隐藏状态，就像直接在模型脑袋里插入芯片，仅限白盒模型）、输出端（对生成文本矫正，说错了就纠错）。
3. 多次增强

6.4 降本增效

去除冗余文本（Prompt 压缩）+ 复用计算结果（KV-Cache 机制）。RAGCache=KV 张量缓存库 + 缓存检索器 +RAG 控制器，把例题当做预制菜存好，遇到类似题型直接拿出来用。

参考文献

[1] 毛玉仁等. 大模型基础. 浙大, 视频 https://www.bilibili.com/video/BV1PB6XYFET2(网站 https://github.com/ZJU-LLMs/Foundations-of-LLMs), 2024.