深度学习自然语言处理第五次作业

李明昕 SY2206124

1 大语言模型（LLM）

1.1 Transformer

Transformer [1] 是一种在自然语言处理和机器学习领域中广泛使用的深度学习模型架构。此模型的主要特点是完全放弃了循环和卷积，而是依赖于注意力机制来处理输入数据。

Transformer 的基础构建模块是自注意力（self-attention）机制，也被称为“缩放的点积注意力”（scaled dot-product attention）。给定一组查询（Q）、键（K）和值（V），自注意力机制的公式可以表示为：

$$\text{Attention}(Q, K, V) = \text{softmax}(\frac{QK^T}{\sqrt{d_k}})V$$

其中，QK^T 表示查询和键的点积，d_k 是键的维度，除以 $\sqrt{d_k}$ 是为了防止点积过大导致 softmax 函数梯度过小。softmax 函数确保了所有的注意力权重都在 0 和 1 之间，并且和为 1，从而使得模型可以在不同的输入之间进行软选择。

一个 Transformer 的编码器包含多个这样的注意力层，每一层都接收所有先前层的输出作为输入。每个 Transformer 编码器还包括一个前馈神经网络层，这是一个完全连接的网络。

Transformer 的解码器也包含多个类似的层，但是额外包含一个第二种类型的注意力机制，这种机制的键和值来自编码器的输出，而查询来自解码器的先前层。这使得解码器可以在生成每个新的输出词元时都考虑到输入序列的所有词元。

这种模型已经在各种自然语言处理任务中取得了出色的性能，包括机器翻译、文本摘要、情感分析等。

目前主流的语言模型有三类：Encoder-Decoder 语言模型、Encoder-Only 语言模型、Decoder-Only 语言模型。其中，Encoder-Decoder 语言模型由两部分组成：编码器（Encoder）和解码器（Decoder），采用这种架构的模型包括谷歌推出的 T5 模型等。Encoder-Only 模型只使用编码器。在这种架构中，注意力的方向是双向的，采用这种架构的模型包括 BERT, RoBERTa 等。Decoder-Only 模型只使用解码器，在这种架构中，注意力的方向是单向的，采用这种架构的模型包括 GPT系列, LLaMA 等。

1.2 指令微调（Instruction Tuning）

指令微调[2]是一种用于优化大型语言模型的技术。这种技术通过使用机器生成的遵循指令的数据进行微调，使得这些模型能够在新任务上展现出显著的零样本（zero-shot）能力。

为了使 LLMs 能够遵循自然语言指令并完成实际任务，研究人员探索了一些方法来对 LLMs 进行指令微调。这通常是通过使用人类注释的提示和反馈在各种任务上微调模型，或者使用公开基准和数据集（这些数据集用手动或自动生成的指令进行了增强）进行有监督的微调来实现的。

在指令微调的基础上，OpenAI 的研究人员还进一步通过强化学习（具体来说称为基于人类反馈的强化学习，RLHF [3]）来进一步对齐模型的输出与人类的偏好（训练得到的模型称为 InstructGPT）。在这一方法中：首先使用指令微调数据集训练一个监督学习模型 SFT；然后训练一个奖励模型 RM；最后使用 RM 对 SFT 进行强化学习得到最终的 PPO 模型。经过 RLHF 训练的模型在人类偏好上体现出了强大的能力。

1.3 上下文学习（In-Context Learning）

上下文学习（In-Context Learning）允许模型基于其之前所见的文本信息来生成响应，这使得它能够参与更自然和连贯的对话，并能够在会话中维持上下文。

在以往的研究中，上下文学习已经显示出了显著的效果，例如在推理、链式思考等任务中。这种方法的一个关键优点是，模型可以在没有任何特定指令或反馈的情况下，仅通过上下文来进行新任务的学习。

1.3.1 思维链（Chain-of-Thought）

思维链的方法旨在让模型先给出推理过程，然后根据推理过程再给出最终的结果。包括零样本（zero-shot）思维链与少样本（few-shot）思维链。

2 实验设置

2.1 模型

本次实验共使用到了三个较新的大语言模型：

1. GPT-3.5: 该模型又称为 ChatGPT，其相关技术与前文所属的 InstructGPT 相近，都是采用 RLHF 得到的；
2. LLaMA-30b-SuperCoT: 该模型基于 Meta 开源的 LLaMA 模型，结合 LoRA 技术进行训练，并在训练中结合思想链数据集、代码解释和说明、片段、逻辑推导等数据；
3. Falcon-40b-Instruct: 该模型基于开源大模型 Falcon，并结合 LoRA 技术进行指令微调。

选择第一个模型的原因是其目前被广泛用于各种评估，且是目前最流行的大模型。选择后两个模型的原因是参考了 Huggingface 的开源大模型排行榜 [4]，而这两个模型是排行榜中较为靠前的两个模型。

2.2 使用方法

按照惯例，在使用大模型进行任务时，将输入给模型的提示（prompt）分为三个部分：

1. 指令（Instruction）：指示模型要进行的任务；
2. 输入（Input）：对任务的补充说明或是任务的上下文。以机器翻译任务为例，这里就提供要翻译的原文；
3. 输出（Output）：针对该任务的期望输出。

不同模型组合这三部分的格式略有不同：对于 GPT-3.5，由于其只能通过调用 API 进行使用，且在 API 中按 "role" 对输入的部分进行了区分，所以我将这三个部分放到了不同 "role" 中，其中指令放在 "system" 中，输入放在 "user" 中，输出放在 "assistant" 中；对于剩下两个模型，由于可以完全掌握上下文的设置，所以直接在上下文中标出这三个部分即可。有关具体设置可以参考 `src\agent.py`（需要注意的是，我还额外在指令中要求返回 JSON 格式的数据）。

在使用模型，还可以分为三个场景：

1. 零样本：在这一使用场景下，只为模型提供**指令**以及**输入**，然后要求模型给出**输出**；
2. 少样本：在这一使用场景下，为模型提供一个**指令**，然后在上下文中提供多个阳历**输入**以及样例**输出**，最后在提供一个测试**输入**，要求模型为最后这个**输入**给出**输出**；
3. 思维链：在这一使用场景下，会在上下文中显示要求模型在给出最终结果前先给出推理过程。

2.3 任务

实验中共包含了三个任务：文本语义相似度、事实问答以及阅读理解。接下来将详细介绍这戏任务的信息，以及为这些任务抽取**指令**，**输入**以及**输出**。

2.3.1 文本语义相似度

文本语义相似度所用的数据集为 STS12 [5]，该任务旨在为两个句子之间的“语义”相似度进行打分，分数取值在 0 到 5 之间。

针对这一数据集，使用以下模板提取 Prompt 中的各个部分：

1. 指令：

```
"Given two sentences, evaluate their semantic similarity on a scale from 0
to 5, where 0 means they are completely dissimilar (the sentences share no
meaningful information in common) and 5 means they are identical or
equivalent in meaning (they essentially express the same information)."
```

2. 输入：

```
"""Sentence 1:
<sentence1>
Sentence 2:
<sentence2>"""
```

3. 输出：

```
"""{
  Score: <score>
}"""
```

评价时使用的指标为 Spearman 相关系数以及 Pearson 相关系数。

2.3.1 事实问答

事实问答所用的数据集为 TruthfulQA [6]，TruthfulQA 是由 Stephanie Lin（牛津大学）、Jacob Hilton（OpenAI）和 Owain Evans（牛津大学）等人提出的一个衡量语言模型是否能生成真实回答的基准测试1。它包含了817个问题，涵盖了38个类别，包括健康、法律、金融和政治等领域。这些问题被设计成一些人可能因为错误的信仰或误解而回答错误。为了在此测试中表现良好，模型必须避免生成从模仿人类文本中学习到的错误答案。

本次实验中，测试的是单选场景下的结果，针对这一数据集，使用以下模板提取 Prompt 中的各个部分：

1. 指令：

```
"Given a question designed to cause imitative falsehoods (false answers),
and multiple candidate answers to the question, determine which candidate
answer is the most truthful."
```

2. 输入：

```
"""Question:
<question>
Candidate Answer 0:
<candidate answer 0>
Candidate Answer 1:
<candidate answer 1>
...
"""
```

3. 输出：

```
"""{
  Sequence: <sequence number>
}"""
```

评价时使用的指标为准确度。

2.3.1 阅读理解

事实问答所用的数据集为 SQuAD [7]，该数据集是一个大规模阅读理解和问题回答数据集。它由斯坦福大学的人工智能实验室发布。包含10万多个问题，这些问题由众包工人在维基百科文章中提出，旨在回答维基百科文章中的内容。问题和答案都是由人类生成的，答案是维基百科文章中的一个文本段落。

针对这一数据集，使用以下模板提取 Prompt 中的各个部分：

1. 指令：

```
"Given a context and a question, extract several consecutive words, as short as possible, from the context as the answer to the question."
```

2. 输入：

```
"""Context:
{context}
Question:
{question}"""
```

3. 输出：

```
"""{
  Answer: <answer>
}"""
```

评价时使用的指标为完全匹配（Exact Match）以及 F1。

3 实验结果

由于本地跑大模型（LLaMA-30b 以及 Falcon-40b）需要大量的计算资源，时间消耗较长，以及调用 GPT-3.5 需要一定花费，所以对每个数据集，都只测试 100 条数据。

此外，由于我采用的方法是让大语言模型输出 JSON 格式的数据，然后再进行解析，所以还可能遇到无法自动解析 JSON 的情况，对于这些数据，该条数据处理出错。

3.1 文本语义相似度（STS12）

3.1.1 零样本

指标	GPT-3.5	LLaMA-30b-SuperCoT	Falcon-40b-Instruct
Spearman	0.7450	0.4696	0.5399
Pearson	0.8853	0.7405	0.7157
自动识别出错个数	1	0	0

3.1.2 少样本

指标	GPT-3.5	LLaMA-30b-SuperCoT	Falcon-40b-Instruct
Spearman	0.7278	0.5699	0.6876
Pearson	0.8695	0.7607	0.8397
自动识别出错个数	0	0	0

3.1.3 思维链

指标	GPT-3.5	LLaMA-30b-SuperCoT	Falcon-40b-Instruct
Spearman	0.7206	0.5960	0.6398
Pearson	0.8662	0.7906	0.7332
自动识别出错个数	0	0	0

从以上三个场景的结果可以看出，在该任务上，模型性能由好到坏排序为：GPT-3.5, Falcon-40b-Instruct, LLaMA-30b-SuperCoT。

3.2 事实问答（TruthfulQA）

3.2.1 零样本

指标	GPT-3.5	LLaMA-30b-SuperCoT	Falcon-40b-Instruct
Accuracy	0.4500	0.5300	0.2700
自动识别出错个数	23	0	0

3.2.2 少样本

指标	GPT-3.5	LLaMA-30b-SuperCoT	Falcon-40b-Instruct
Accuracy	0.6400	0.8900	-
自动识别出错个数	0	0	-

3.2.3 思维链

指标	GPT-3.5	LLaMA-30b-SuperCoT	Falcon-40b-Instruct
Accuracy	0.3900	0.5900	0.2700
自动识别出错个数	45	0	0

从以上三个场景的结果可以看出，在该任务上，模型性能由好到坏排序为：LLaMA-30b-SuperCoT, GPT-3.5, Falcon-40b-Instruct。但在这个任务中，LLaMA 模型的性能过于突出，所以不能排除其训练集中有用到 TruthfulQA 数据的情况。

3.3 阅读理解（SQuAD）

3.3.1 零样本

指标	GPT-3.5	LLaMA-30b-SuperCoT	Falcon-40b-Instruct
Exact Match	0.4900	0.6900	0.6600
F1	0.7029	0.8104	0.8046
自动识别出错个数	11	2	0

3.3.2 少样本

由于模板设置的问题，导致这部分自动识别出现了问题，所以暂时不统计这部分信息。

3.3.3 思维链

指标	GPT-3.5	LLaMA-30b-SuperCoT	Falcon-40b-Instruct
Exact Match	0.6400	0.8000	0.4400
F1	0.8135	0.8763	0.6252
自动识别出错个数	2	2	2

从以上两个场景的结果可以看出，在该任务上，模型性能由好到坏排序为：LLaMA-30b-SuperCoT, GPT-3.5, Falcon-40b-Instruct。

从所有任务的结果来看，LLaMA-30b-SuperCoT 模型也具有相当优秀的性能。当然，由于实际使用到的数据量并不多，所以这里的结果也不具有显著的代表性，但仍然能观察到一下特点：

1. 所有模型的少样本性能基本都能好于零样本性能，并且少样本场景下还能介绍自动识别出错的概率，说明少样本方法在实际使用场景中的优越性；
2. 相较于其他模型，LLaMA-30b-SuperCoT 在零样本思维链场景下的性能会明显优于不使用思维链场景下的性能，而其他模型在这两个场景下的性能相差不多（GPT-3.5只在一个任务上思维链性能明显好于零样本），有时甚至会出现性能倒退。可能的原因是 LLaMA-30b-SuperCoT 专门在思维链数据中进行了训练。

3.4 讨论

在使用大模型进行下游任务时，其优势主要在于只需要很少的数据，就能将模型应用到对应任务，减少了部署过程对数据的依赖。

但与此同时仍然存在很多问题需要解决：第一个问题是部署所需的计算资源较大，以 LLaMA-30b-SuperCoT 为例，即使使用了 4bit 量化技术，要在一张卡上使用该模型进行推理也需要大概 20g 的显存，并且模型的推理速度也比较慢，这些就会阻碍模型在实际应用场景下的部署；第二个问题是提示（prompt）的设计问题，虽然本次实验中没有针对这一方向的实验，但已经有很多研究发现，大模型在不同模板下表现出的性能可能相差很大，这就导致有时候需要消耗很多精力进行提示（prompt）的设计；第三个问题是性能上的问题，虽然大语言模型加少样本学习能达到比较好的性能，但实际上该性能与完全在该任务下进行微调的 SOTA 模型的性能还有一定差距。当然，这两者在所需的数据量上存在明显差异，但如果实际的应用场景非常关注模型的性能，这种性能上的差距就会让大模型的部署陷入窘境。

最后要说的是，大模型由于其强大的零样本、少样本学习能力，其不应只被应用于各种下游任务，我们应该更深入地发掘各种应用场景以激发其潜能。

参考文献

[1] Vaswani A, Shazeer N, Parmar N, et al. Attention is all you need[J]. Advances in neural information processing systems, 2017, 30.

[2] Wei J, Bosma M, Zhao V Y, et al. Finetuned language models are zero-shot learners[J]. arXiv preprint arXiv:2109.01652, 2021.

[3] Ouyang L, Wu J, Jiang X, et al. Training language models to follow instructions with human feedback[J]. Advances in Neural Information Processing Systems, 2022, 35: 27730-27744.

[4] Huggingface. Open LLM Leaderboard[EB/OL]. (2023-06)[2023-06-08]. https://huggingface.co/spaces/HuggingFaceH4/open_llm_leaderboard

[5] Agirre E, Cer D, Diab M, et al. Semeval-2012 task 6: A pilot on semantic textual similarity[C]//* SEM 2012: The First Joint Conference on Lexical and Computational Semantics–Volume 1: Proceedings of the main conference and the shared task, and Volume 2: Proceedings of the Sixth International Workshop on Semantic Evaluation (SemEval 2012). 2012: 385-393.

[6] Lin S, Hilton J, Evans O. Truthfulqa: Measuring how models mimic human falsehoods[J]. arXiv preprint arXiv:2109.07958, 2021.

[7] Rajpurkar P, Zhang J, Lopyrev K, et al. Squad: 100,000+ questions for machine comprehension of text[J]. arXiv preprint arXiv:1606.05250, 2016.