规则组合实现大语言模型的数字理解能力

陈钶杰
学号: 22135030

January 30, 2024

1 引言(introduction)

自然语言处理领域的最新进展中，大语言模型如GPT-4已经取得了显著的成功，展示了在各种任务中惊人的语言理解能力。然而，在处理数字问题等专业领域时，这些模型仍然面临挑战,这些模型在理解数字时,主要通过背诵和记忆来理解,而不是理解数字之间本身l的规则。本文研究旨在深入探讨大语言模型在关于数字上的规则学习，以提高模型在这一领域的一些性能。

2 相关工作(relate work)

2.1 背景介绍(back ground)

大语言模型通过大规模数据学习语言规则，但在数学计算中，数字之间的运算是无穷无尽的，但是数字计算是有明确规则的,数字运算对于模型而言是一项具有挑战性的任务。当前的语言模型在这方面仍然存在一些限制，因此有必要进一步研究数字规则学习的方法，以增强其在数字运算中的适用性。

2.2 先前研究综述

早期的已经有不少研究集中在使用大型语言模型在解决数字计算。比如ComputeGPT[3]通过借助python工具来进行数字的计算,这种方法的好处是计算能力强大,且计算的数值不受任何位数的影响,但是这种方法语言模型只是起到了一个文本识别器,将数字,运算符号识别出来,真正文献进行运算的是python,而语言模型本身是不会进行数字计算的.像程序辅助语言模型[1]也是类似的,这是一种新的自然语言推理方法，使用程序作为中间推理步骤。与现有的基于LLM的推理方法不同，主要思想是将求解和计算offload到外部Python 解释器，而不是使用LLM 来理解问题和求解。像MetaMath[5],让语言模型去执行算术和符号推理的任务,通过大量计算题数据的训练,来实现对一些问题的计算,但这种方式更像是背题,而不是深刻理解其中数学规律和算法.MathGPT[4]是一种大型语言模型在不使用计算器工具的情况下无法准确执行算术运算，在足够的训练数据下，20亿参数的语言模型可以准确地进行多位算术运算，此文的核心思路是尽可能的学习5位数以内的数学计算规则,通过这些规则的学习,对高位数运算进行学习.与利用外部工具不同，我们专注于探索如何在不依赖外部工具的情况下增强llm固有的算术能力,但是这个模型并没有让模型真正理解数字运算的基本规律.

2.3 比较和对比

我们的模型同样专注于探索如何在不依赖外部工具的情况下增强llm固有的算术能力,但不同之处在于我们更注重语言模型如何去理解数学计算,而不是通过背诵的方式来解决数学问题.比如对于加法,我们的模型只需要学会99加法表以及加法的进位,对齐规则就可以计算任意位数的加法.这种从加法的数学出发进行计算不仅能够保证计算的准确率,而且更加体现神经网络的泛化能力.

2.4 研究趋势和发展

MLC[2]说明神经网络模型是具有规则学习能力,且具有泛化能力,而目前语言模型在数字计算主流解决方法都是通过采用大量数据训练来提高准确性,其中通过优化训练数据提高最终模型的准确性.

2.5 未解决的问题

通过大规模数据的训练这样虽然能让语言模型掌握一定数学计算能力,但是对于其中计算具有很强的随机性,即使通过使用高质量数据能对模型性能有一定的提升,但是很难有质的改变.因此我们希望能够让语言模型通过学习数学规则来提高数字计算的准确性.只要学会规则,那么模型就能够以极高的准确率去解决各种计算问题.这样是非常符合我们预期的.

2.6 方法和技术的演进

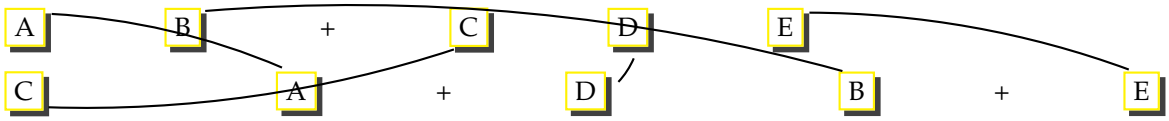
transformer模型在文本处理上依旧是最优秀的,我们使用基本的transformer模型框架,将规则分为基本数学计算规则以及进位,对齐等复合规则,同时进行训练,最终模型在经过训练以后测试其数字运算的能力.

3 Methodology

传统的语言模型进行数值计算的本质是回忆，比如计算12 + 23 和计算23 + 12是两种完全不同的计算思路，任意两位数相加都当成是一种单独的回忆，这样一来，两位数需要记忆10000种情况,一旦位数变高,训练的数据量就会变得巨大.然而这些计算其实都是可以按照固定的规则来做,这样不仅能减少训练数据量,且更加符合数学计算的本质.我们希望模型通过遵循规则的方式,即通过制定一系列子规则和复合规则来进行计算求解,数值问题.在我们的规则中,子规则是10以内的数之间的各个加法,比如1+2 = 3，3+4 = 7等等,而复合规则包括对齐和进位等.我们只需要使用一个基本规则就能解决一系列问题.比如三位数以内的对齐规则如下图所示.原本需要10000个数据集，现在我们可以通过只用4个简单规则来解决这些问题。假设n位数进行计算，那么两者的比值大约是$\frac{10^n}{n^2}$,当n很大时，会有明显的效率提升.经过对齐以后可以利用已经学得的子规则进行处理.数值计算往往是需要进位的，比如计算33+77时,我们需要进位,而这个进位也可以通过一些简单的规则来实现.如下图所示规则进行进位了,最终即可得到结果.

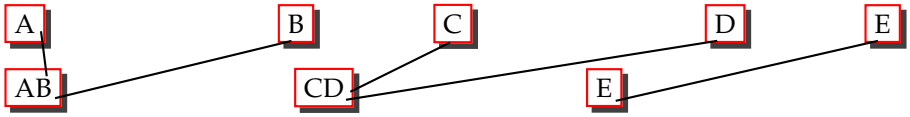
3.1 mapping regular

3.2 align regular:



3.3 compute regular:

3.4 carry regular:



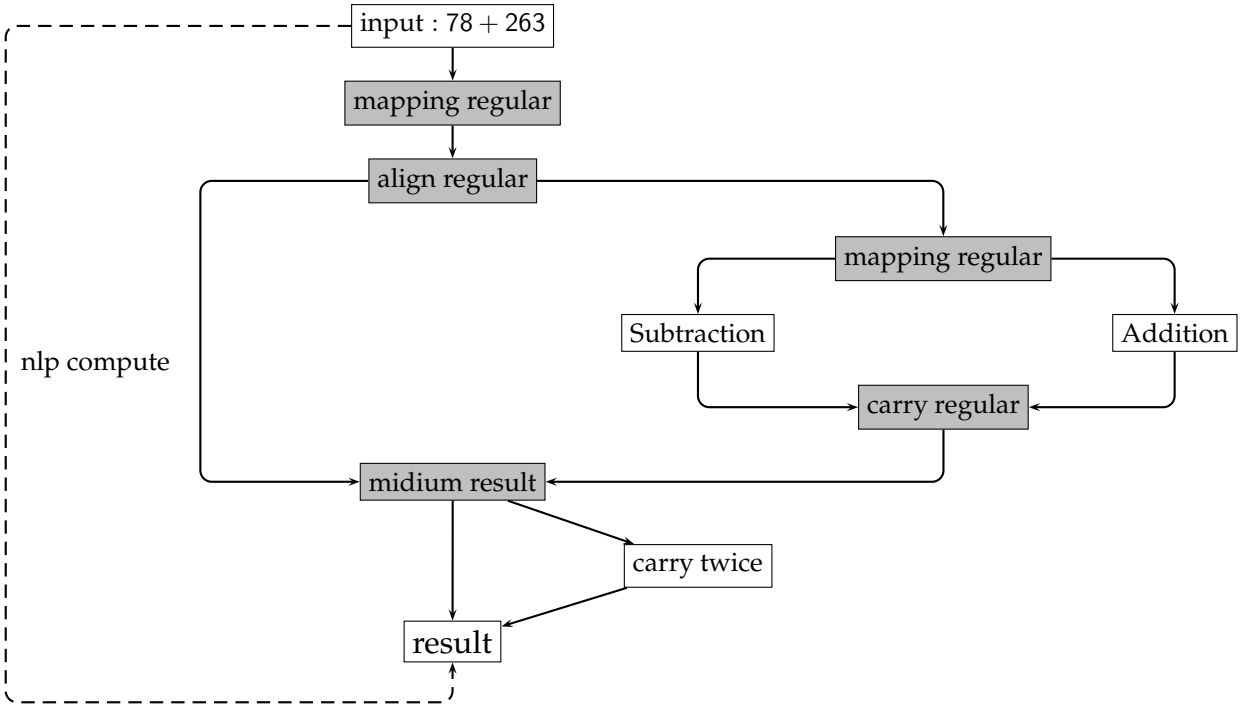


Figure 1: 如何调用规则求解算式

3.5 模型框架

4 experiments

使用5位数的所有计算结果。

4.1 针对别人计算不准确的问题进行比较,比如做一些减法问题

4.2 使用公认的数学计算数据集进行测试

4.3 对特殊问题进行分类，比如小数减大数

References

[1] Luyu Gao, Aman Madaan, Shuyan Zhou, Uri Alon, Pengfei Liu, Yiming Yang, Jamie Callan, and Graham Neubig. Pal: Program-aided language models. *arXiv preprint arXiv:2211.10435*, 2022.

[2] Brenden M Lake and Marco Baroni. Human-like systematic generalization through a meta-learning neural network. *Nature*, 623(7985):115–121, 2023.

[3] Ryan Hardesty Lewis and Junfeng Jiao. Computegpt: A computational chat model for numerical problems. *arXiv preprint arXiv:2305.06223*, 2023.

[4] Zhen Yang, Ming Ding, Qingsong Lv, Zhihuan Jiang, Zehai He, Yuyi Guo, Jinfeng Bai, and Jie Tang. Gpt can solve mathematical problems without a calculator. *arXiv preprint arXiv:2309.03241*, 2023.

[5] Longhui Yu, Weisen Jiang, Han Shi, Jincheng Yu, Zhengying Liu, Yu Zhang, James T Kwok, Zhenguo Li, Adrian Weller, and Weiyang Liu. Metamath: Bootstrap your own mathematical questions for large language models. *arXiv preprint arXiv:2309.12284*, 2023.