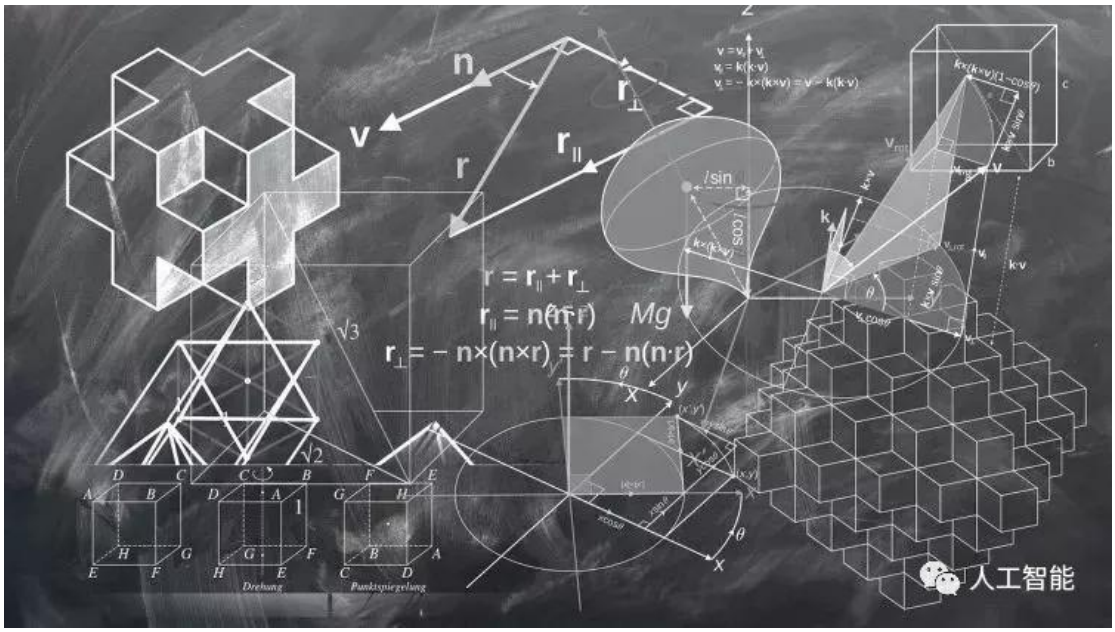
战胜柯洁战胜不了高中生？DeepMind 挑战高中数学题，完败

人工智能4 月 8 日

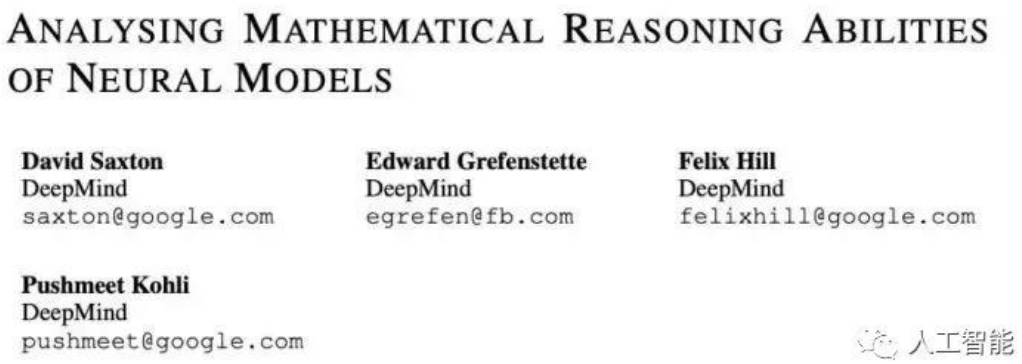
被数学题难倒的 AI。



做数学题一直令多数人头疼不已的事情。近期，DeepMind 团队最新研究了利用 AI 来解数学题，但结果令人大跌眼镜——水平不及高中生。

数学也难倒了 AI。

数学可能是大多数人在求学过程中最头疼的一门科目。近日 ,DeepMind 团队便对“AI 做数学题”进行了研究，结果大跌眼镜：“万能的 AI”在面对数学问题也是不知所措！



ANALYSING MATHEMATICAL REASONING ABILITIES OF NEURAL MODELS

David Saxton
DeepMind
saxton@google.com

Edward Grefenstette
DeepMind
egrefen@fb.com

Felix Hill
DeepMind
felixhill@google.com

Pushmeet Kohli
DeepMind
pushmeet@google.com

人类解题能力超群的关键在于，人类并非主要通过经验和证据，而是通过推断、学习，以及利用定理、公理和符号操纵规则。

DeepMind 团队便对神经架构和类似系统的评估(以及最终的设计)提出了新的挑战，开发了一个数学问题的任务处理套件，涉及以自由形式文本输入/输出格式的系列问题和答案。

不过，在研究过程中，DeepMind 发现，AI 非常擅长做的数学题都是比较偏简单的，例如：查找数字中的位值、四舍五入小数/整数等。但是在诸如素数检测、因式分解以及多项式操作等方面，性能结果存在显著的差异。

AI 做数学的能力不及高中生水平？

AI 挑战人类最难学科

深层模型远未达到人类所表现出的稳健性和灵活性，由于自身能力的限制，深度学习无法超越所经历的环境去生成新的东西，并且面对存在对抗性构建的输入时极其脆弱。

与神经模型相比，人类智能擅长的一个领域是关于物体和实体的离散组合推理，即“代数泛化”，这个领域也体现了神经模型和人类智之间的差异。

人类在这个领域内的概括能力是复杂的、多方面的。先来看这个数学题：

当：f(x)= 2x + 3，g(x)= 7x-4，h(x)= -5x-8 时

求：g(h(f(x)))

人类解决这道数学题时候，用到的各种认知技能有：

将字符解析为数字，算术运算符，变量（一起形成函数）和单词（确定问题）等实体

计划（例如，以正确的顺序识别功能以进行撰写）

使用子算法进行函数合成（加法，乘法）

利用工作记忆来存储中间值（例如合成 h(f(x)) ）

通常应用已获得的规则，转换，过程和公理知识

DeepMind 在这篇论文中引入了一个由许多不同类型的数学问题组成的数据集 ,对于模型来说，优于缺乏上述人类能力，在处理跨系列的问题类型（包括我们在下面详述的泛化）的时候难度更大，更难获得良好的表现。

该领域对于一般的神经结构的分析是重要的。除了提供广泛的问题外，还有其他几个优点：

数学提供了一个自洽的宇宙(self-consistent universe)；

符号在不同的问题类型中是相同的，是的数据集更容易得到扩展的；

在一种问题类型上学习的规则和方法通常适用于其他地方。例如数字的加法在任何地方都遵循相同的规则，并且在其他问题中作为“子程序”出现，具体体现在乘法中，以及具体且更抽闲的体现在多项式中；

具有转移知识能力的模型将在数据集上获得更好的表现（知识迁移可能是解决更难问题的必要条件）。

数学本身也是一个有趣的领域，虽然解决该数据集中大多数中学数学问题的模型本身不具备应用程序，但它们可能会导致更强大的模型，这些模型可以解决有趣且实质性的新数学问题。

或者更一般地说，寻求验证以捕获算法/系统推理为目标的新架构的实验经常从这个领域中得出，这并非巧合。因此，在为这些模型提供大规模的训练和评估框架时，希望为继续研究超越数学的机器推理提供坚实的基础。

请看以下数学问题集示例：

问题：对于 r，求解-42*r+27*c=-1167 和 130*r+4*c=372。

答案：4

问题：计算-841880142.544+411127。

答案：-841469015.544

问题：Letx(g)=9*g+1。Letq(C)=2*C+1。Letf(i)=3*i-39. 设 w(j)=q(x(j))。计算 f(w(a))。

答案：54*a-30

问题：设 e(l)=l-6.2 是 e(9)和 2 的因子吗？

答案：错

问题：设 u(n)=-n**3-n**2。设 e(c)=-2*c**3+c。令 l(j)=-118*e(j)+54*u(j)。l(a)的衍生物是什么？

答案：546*a**2-108*a-118

问题：从 qqqkkklkqkkk 中选择了三个字母而没有替换。给出序列 qql 的概率

答案：1/110

研究中的主要贡献

数据集和泛化测试

研究人员发布 1 个序列到序列的数据集，包括许多不同类型的数学问题（见图 1），用于测量数学推理，同时提供生成代码和预生成的问题。

数据集附带两组测试：插值测试，一个针对训练集中出现的每种类型的问题；外推测试，测量沿着各种难度轴的概括超出训练期间的概括。将外推测试作为模型是否采用允许它们进行代数泛化的能力的额外度量。

实验和模型分析

本文利用一个实验评估来研究最先进的神经架构的代数能力，实验表明它们在某些类型的问题上表现良好，但肯定不是全部，而且只有适度的数量一般化。我们对他们如何学习回答数学问题及其失败模式提供了一些见解。

由于该数据集背后的构建过程，有大量现有模型可以进行调整、专门构建或定制，以解决提出的问题，特别是在符号求解器或计算机代数系统的帮助下。

模型检验

随着问题和答案的复杂性或语言多样性的增长，撇开传统符号方法可能的脆弱性或可扩展性的限制，我们对评估通用模型更感兴趣，而非已经内置数学知识的模型。

使这些模型（总是神经架构）从翻译到通过图像字幕解析无处不在的原因，是这些函数逼近器缺乏偏差，因为它们的设计中编码的域特定知识相对较少（或没有）。

虽然有一些神经网络驱动的方法可以直接访问数学运算（例如加法或乘法，或更复杂的数学模板，这无疑是我们在本文中提出的任务中具有竞争力，我们将局限于一般的序列处理架构，这些架构用于其他非数学任务，以便为将来的比较提供最一般的基准。

论文研究了两种（广泛的）模型，这些模型已经证明了它们在序列到序列问题上的最新技术：循环神经架构，以及最近引入的 Attention/Transfomer 结构。 我们还尝试使用可微分神经计算机，这是一种具有“外部存储器”的复现模型（其大小与网络中的参数数量无关）。

理论上，这可能非常适合解决数学问题，因为它可以存储中间值以供以后使用。然而，却无法从中获得不错的表现，即使对于内存插槽的数量和大小的超参数扫描等，在训练一天后才能达到 10％的验证性能，而大多数模型在不到一个小时内就能获得这一点。

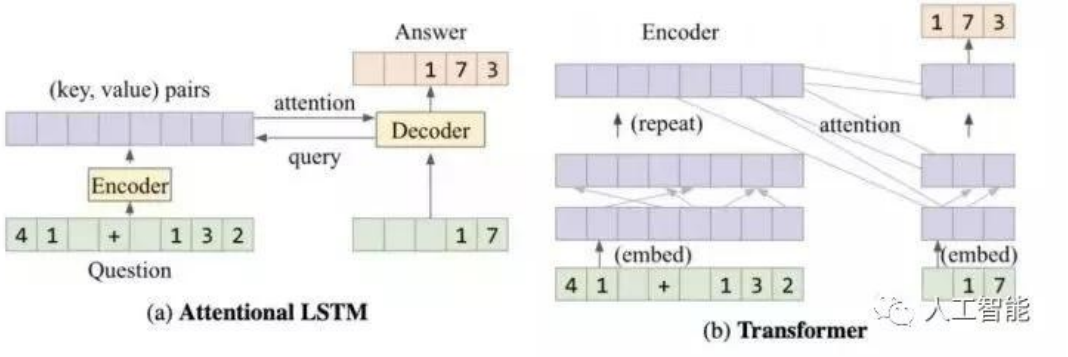


图 2 :注意力 LSTM 和 Transformer 体系结构都包含一个解析问题的编码器和一个解码器，它将正确的答案右移 1 个映射到每个位置的答案中的下一个字符（因此允许自回归预测）：

(a)注意 LSTM 将问题编码为一系列（关键，值）位置，然后由解码器进行处理

(b)变压器有几个阶段的自我注意和输入注意

循环结构

LSTM 是一个强大的序列到序列模型构建模块，它在许多领域都达到了最先进的结果，尽管它很简单，但仍然是循环神经网络的一个核心构建模块。本文测试了两个标准的循环结构。

第一个(也是最简单)模型，称作“Simple LSTM”是直接将问题提交到 LSTM，一次输入一个字符(采用 1-hot 编码)；

第二个模型称作“Attentionnal LSTM”，是引入具有注意力结构的编码器/解码器。

在这两种体系结构中，还使用了一个简单的更改来提高性能。所描述的模型必须在解析问题之后直接输出答案。

近期，一种称为关系递归神经网络或关系内存核(relational memory core，RMC)的递归体系结构被开发出来作为 LSTM 的替代品。这个重复单元有多个记忆槽，它们通过注意力相互作用。

TRANSFORMER

Transformer 模型是一个实现机器翻译的最先进结果的序列到序列模型。图 2b 对其做了简要的描述。该模型由编码器和解码器组成，前者将问题(表示为向量序列)转换为另一个相同长度的序列，后者将编码的问题和答案转换为答案预测。

性能分析

训练和评估方法

与序列到序列模型中常见的方法一样，这些模型使用贪婪解码器(每一步输出多数类)自回归地预测答案。通过 Adam 优化器最小化正确字符的对数概率之和，学习率为 6×10-4 ,β1= 0.9 ,β2= 0.995 ,ε= 10-9。使用批量大小为 1024 的 8 个 NVIDIA P100 GPU 进行 500k 批次分割，绝对梯度值限幅为 0.1。

实验结果

图 3 显示了不同结构的平均插值和外推(extrapolation)性能。

	Parameters	Interpolation	Extrapolation
Simple **LSTM**	18M	0.57	0.41
Simple **RMC**	38M	0.53	0.38
Attentional **LSTM**, LSTM encoder	24M	0.57	0.38
Attentional **LSTM**, bidir LSTM encoder	26M	0.58	0.42
Attentional **RMC**, bidir LSTM encoder	39M	0.54	0.43
Transformer	30M	**0.76**	**0.50**

图 3 模型精度(正确答案的概率)在各个模块之间取平均值。RMC 是关系递归神经网络模型。

LSTMs vs RMCs

使用具有多个内存插槽的 RMC 不会提高性能；也许 RMC 很难学会使用插槽来操纵数学实体。对于给定数量的隐含单元，RMC 的数据效率更高，但训练速度更慢(因为它们有更多的参数)，LSTMs 具有更好的渐近性能。

Simple vs Attentional LSTM

Attentional LSTM 和 Simple LSTM 具有相似的性能。有人可能会怀疑 Attentional LSTM 什么也不做，但事实并非如此，因为与解析 LSTM 大小相同的 Simple LSTM 模型获得的性能要差得多。我们推测，注意力模型并没有学习算法解析问题，因此每一步改变注意力焦点的能力并不重要。

“思考”步骤数

对于 Attentional LSTM 模型，可以观察到，将“思考”步骤的数量从 0 增加到 16，可以提高性能。

Transformer vs 最好的非 transformer 模型

Transformer 在几乎所有模块上的性能与递归模型相同，或者明显优于递归模型。这两种体系结构具有相当数量的参数。人们可能会预先期望 LSTM 执行得更好，因为它的顺序体系结构可能更类似于人类执行的顺序推理步骤。然而，实验表明，这两种网络都没有做太多的“算法推理”，并且 Transformer 相对于 LSTM 架构具有各种优势，例如：

使用相同数量的参数进行更多计算；

具有更好的梯度传播；

有一个内部连续的“记忆”。

对神经网络来说最简单的数学问题

最简单的问题类型是查找数字中的位值，以及四舍五入小数和整数，所有模型在这些方面都获得了近乎完美的分数。涉及比较的问题也往往相当容易，因为这类任务是相当感性的(例如比较长度或单个数字)。

对神经网络来说最困难的数学问题

也许并不奇怪，一些最难的模块包含了更多的数字理论问题，这些问题对人类来说也很难，比如检测素数和因式分解。

Transformer 模型在“加或减几个数字”模块和“乘数或除数”模块的性能为 90％或更高。然而，在混合算术模块上，性能下降到大约 50％。我们推测这些模块之间的区别在于前者可以在相对线性/浅/平行的方式（因此解决方法通过梯度下降相对容易发现），而没有用括号评估混合算术表达式的快捷方式，其中需要计算中间值。

这证明模型没有学习任何代数/算法操作值，而是学习相对简单的技巧来获得许多模块的良好答案。对于其他需要中间值计算的模块，如多项式求值和一般组合，也是如此。

多项式操纵性能

Transformer 和递归模型之间的一个显着差异是多项式操作。Transformer 在多项式展开、收集项、加法、组合、微分和提取命名系数方面做得明显更好。从理论上说，Transformer 的并行顺序特性更擅长于处理多项式，其中几个系数必须同时保存在内存中，以便相互作用。

论文地址：

https://arxiv.org/pdf/1904.01557.pdf