



## 第34回 全脳アーキテクチャ勉強会

### 算術課題における外挿能力の測定

2022年5月11日

株式会社アラヤ

藤澤 逸平

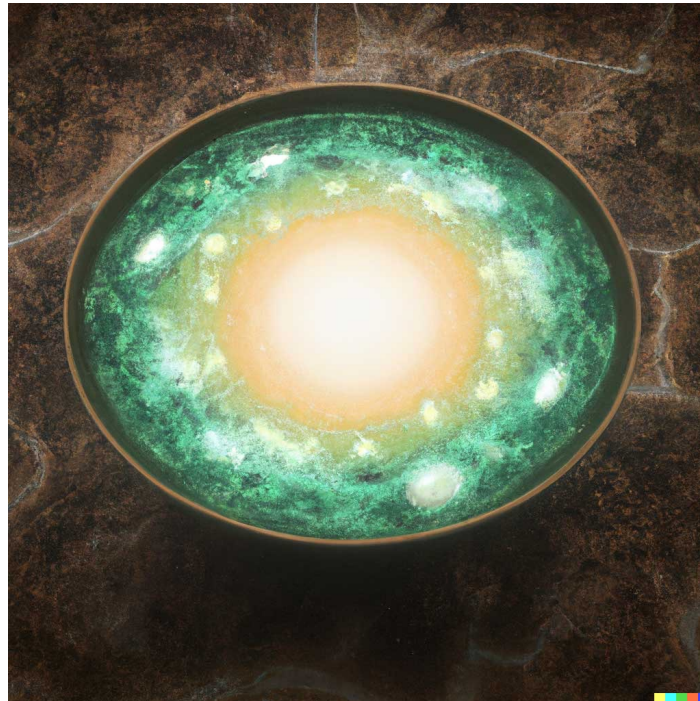


# DALL·E 2

DALL·E 2 is a new AI system that can create realistic images and art from a description in natural language.

[↗ JOIN WAITLIST](#)[↓ EXPLORE](#)[▶ WATCH VIDEO](#)[📄 VIEW RESEARCH](#)[📷 FOLLOW ON INSTAGRAM](#)

## A bowl of soup that is a portal to another dimension as digital art



# OpenAI Codex

August 10, 2021  
3 minute read

We've created an improved version of OpenAI Codex, our AI system that translates natural language to code, and we are releasing it through our API in private beta starting today. Codex is the model that powers [GitHub Copilot](#), which we built and launched in partnership with GitHub a month ago. Proficient in more than a dozen programming languages, Codex can now interpret simple commands in natural language and execute them on the user's behalf—making it possible to build a natural language interface to existing applications. We are now inviting businesses and developers to build on top of OpenAI Codex through our API.

[JOIN THE WAITLIST](#)

[▶ REWATCH LIVE DEMO](#)

[○ VIEW THE CODEX CHALLENGE](#)

[📄 READ PAPER](#)



## Playground

Explain code



Save

```
1 class Log:
2     def __init__(self, path):
3         dirname = os.path.dirname(path)
4         os.makedirs(dirname, exist_ok=True)
5         f = open(path, "a+")
6
7         # Check that the file is newline-terminated
8         size = os.path.getsize(path)
9         if size > 0:
10            f.seek(size - 1)
11            end = f.read(1)
12            if end != "\n":
13                f.write("\n")
14            self.f = f
15            self.path = path
16
17     def log(self, event):
18         event["_event_id"] = str(uuid.uuid4())
19         json.dump(event, self.f)
20         self.f.write("\n")
21
22     def state(self):
23         state = {"complete": set(), "last": None}
24         for line in open(self.path):
25             event = json.loads(line)
26             if event["type"] == "submit" and event["success"]:
27                 state["complete"].add(event["id"])
28                 state["last"] = event
29         return state
30
31 """
32 Here's what the above class is doing:
33 1.
```

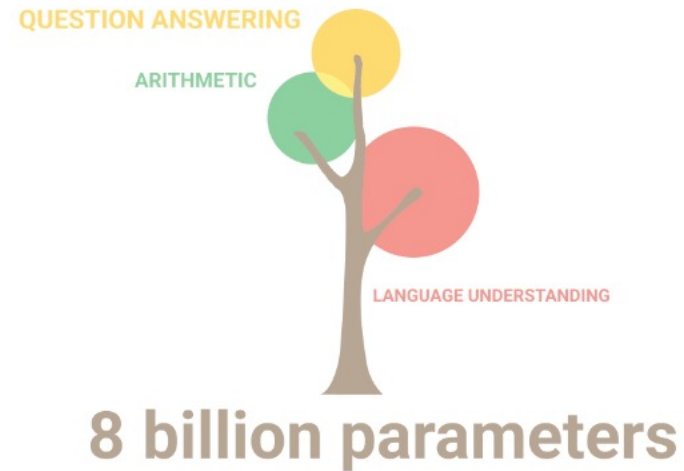
Submit



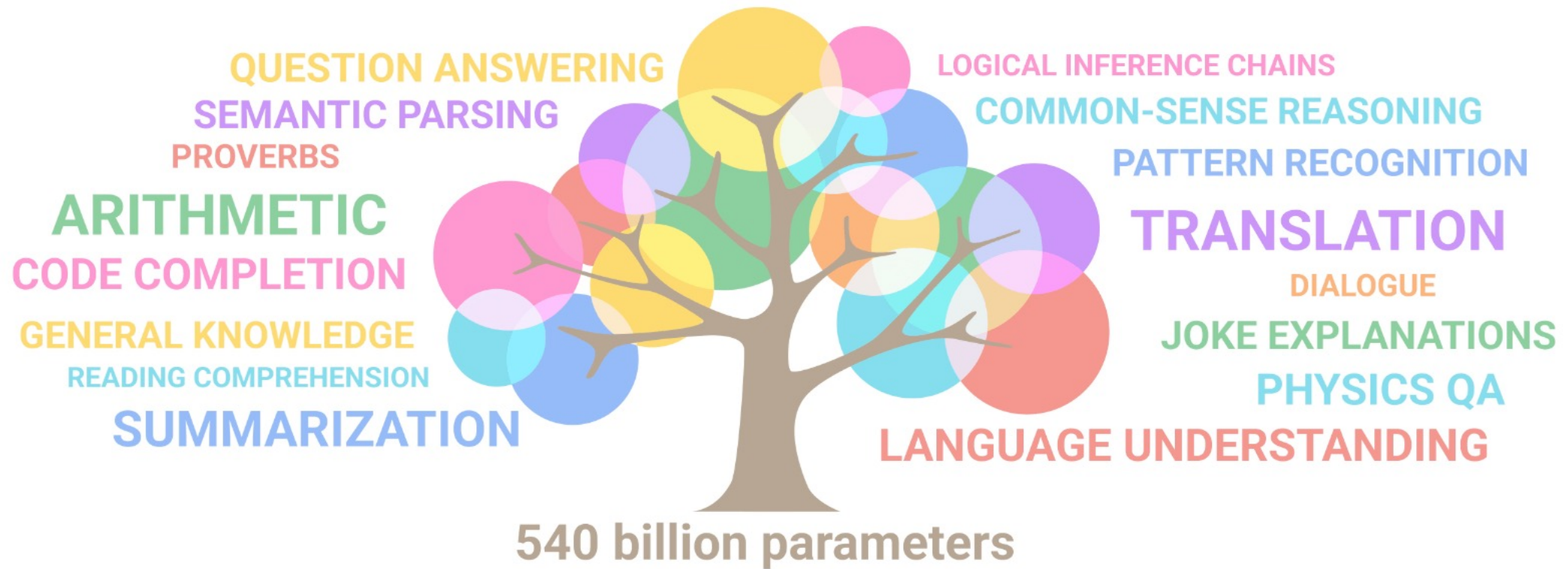
294

Plain text

# Pathways Language Model (PaLM)



# Pathways Language Model (PaLM)



**Input:** Jennifer looked out her window and sees a really cool cloud below her. She unbuckles her seatbelt and heads to the bathroom. Is Jennifer probably traveling more than 300 miles per hour relative to the earth?

**Model Output:** 300 miles per hour is about 480 km/h. This is about the speed of a commercial airplane. Clouds are usually below airplanes, so Jennifer is probably on an airplane. The answer is "yes".

**Input:** ジェニファーが窓の外を見ると、眼下にとっても素敵な雲が広がっています。彼女はシートベルトを外し、トイレに向かいます。ジェニファーはおそらく地球に対して時速300マイル以上走っているのだろうか？

**Model Output:** 時速300マイルは時速約480kmです。これは民間航空機の速度とほぼ同じです。雲は通常、飛行機の下にあるので、ジェニファーはおそらく飛行機に乗っている。答えは「はい」である。

(DeepLで翻訳)



- 深層学習で、できることは増えてきた。
- 一部のタスクでは人間をも凌駕している。
  
- しかし、できないこともいろいろある。
- その中から、次に解けそうな課題について考えたい。
  
- 算術課題やその類を解くことを考えると、次のアーキテクチャ開発につながるのではないか。

- 深層学習で、できることは増えてきた。
  - 一部のタスクでは人間をも凌駕している。
  
  - しかし、できないこともいろいろある。
  - その中から、次に解けそうな課題について考えたい。
  
  - 算術課題やその類を解くことを考えると、次のアーキテクチャ開発につながるのではないか。
1. 深層学習では何ができないか？
  2. 何故、算術課題か？
  3. 算術課題とは、どんな問題設定なのか？
  4. どんなシステムを開発できたら嬉しいのか？
  5. 算術課題周辺で、これまでにどんな研究がなされているのか？
  6. どんな方法で解けそうか？

- “意味”を扱えない
  - “意味”を分かって予測しているわけではない
- 予測の不安定さ
  - 人間による予測のようなロバスト性がない
- 筋の良い外挿ができない
- 訓練には大量のデータが必要
- 目的を自分で設定できない
  - あとから発生したタスクに対応できない
- 複数の目的をうまくバランスさせることが難しい
- 過去に学んだ知識の再利用が難しい
- 明示的な記憶構造がない
- 他のモデルの経験を流用することが難しい

- 近年の深層学習の発展の帰結の1つ「大規模なモデルを大量のデータで訓練すると高精度」
- GPT-3は1,000億以上のパラメタ数。訓練データは570GBを超える。

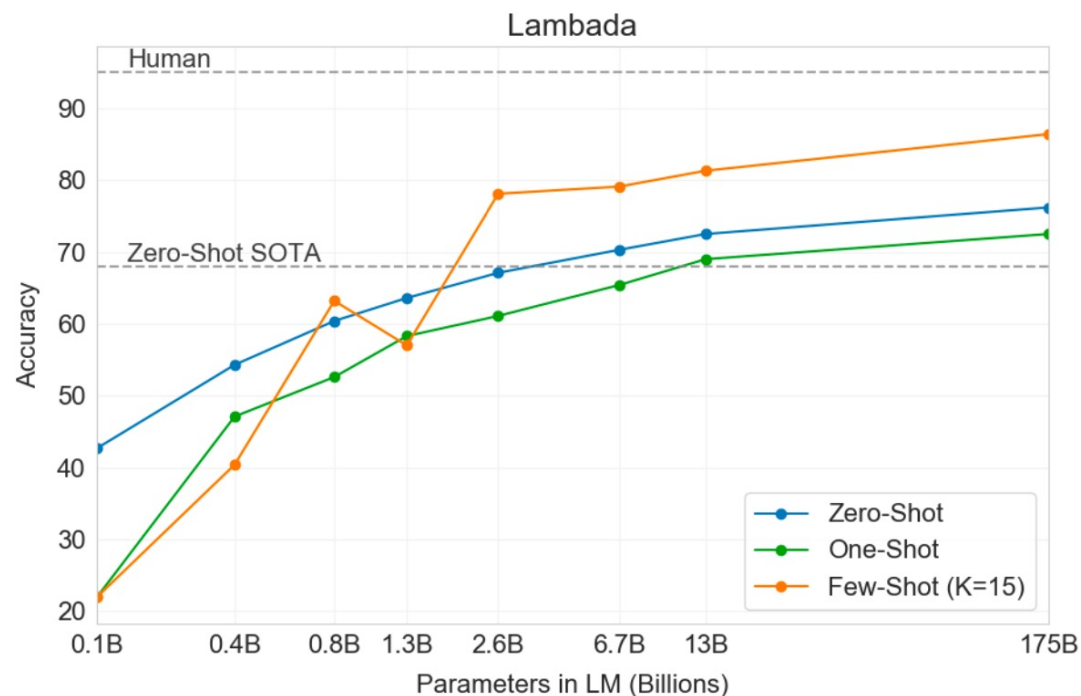
- *We then generated completions of these titles and subtitles from for language models ranging in size from **125M to 175B (GPT-3) parameters** (mean length: 200 words).*
- *[...]*
- *Table 2.2 shows the final mixture of datasets that we used [...] and **570GB after filtering**, roughly equivalent to **400 billion byte-pair-encoded tokens**.*

Dataset	Quantity (tokens)	Weight in training mix	Epochs elapsed when training for 300B tokens
Common Crawl (filtered)	410 billion	60%	0.44
WebText2	19 billion	22%	2.9
Books1	12 billion	8%	1.9
Books2	55 billion	8%	0.43
Wikipedia	3 billion	3%	3.4

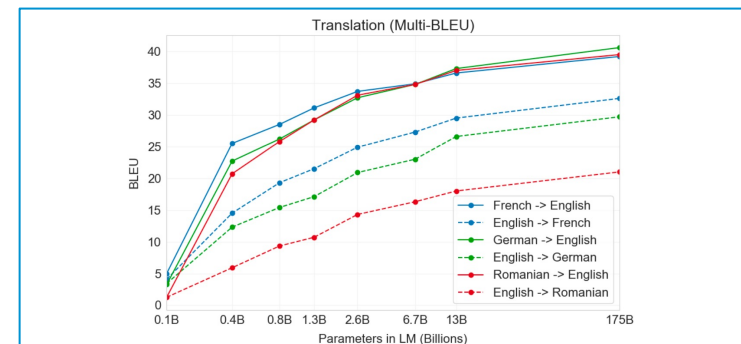
**Table 2.2: Datasets used to train GPT-3.** “Weight in training mix” refers to the fraction of examples during training that are drawn from a given dataset, which we intentionally do not make proportional to the size of the dataset. As a result, when we train for 300 billion tokens, some datasets are seen up to 3.4 times during training while other datasets are seen less than once.

# データとモデルの大規模化 (GPT-3の精度)

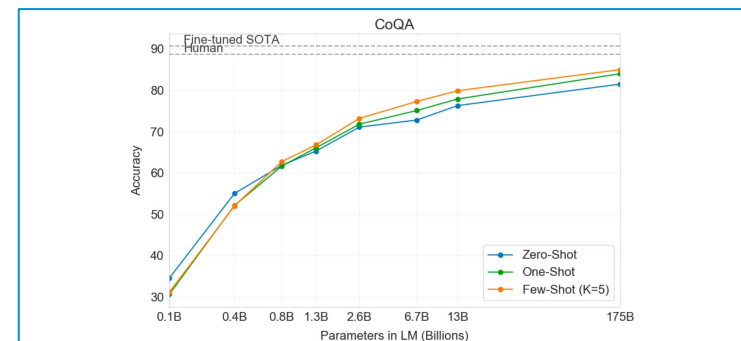
- モデルのパラメタ数を増やすことで、精度が向上する。



**Figure 3.2:** On LAMBADA, the few-shot capability of language models results in a strong boost to accuracy. GPT-3 2.7B outperforms the SOTA 17B parameter Turing-NLG [Tur20] in this setting, and GPT-3 175B advances the state of the art by 18%. Note zero-shot uses a different format from one-shot and few-shot as described in the text.

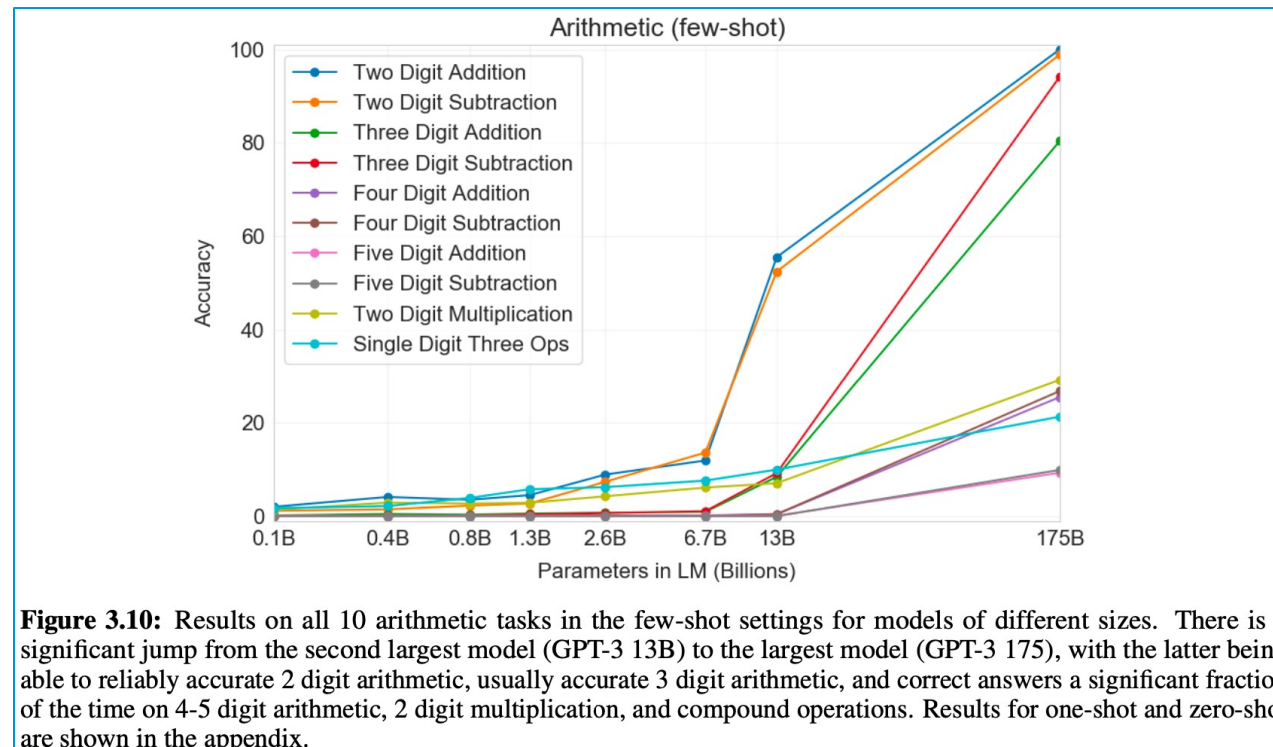


**Figure 3.4:** Few-shot translation performance on 6 language pairs as model capacity increases. There is a consistent trend of improvement across all datasets as the model scales, and as well as tendency for translation into English to be stronger than translation from English.



**Figure 3.7:** GPT-3 results on CoQA reading comprehension task. GPT-3 175B achieves 85 F1 in the few-shot setting, only a few points behind measured human performance and state-of-the-art fine-tuned models. Zero-shot and one-shot performance is a few points behind, with the gains to few-shot being largest for bigger models.

- GPT-3の論文では、10種類の初等算術に対して、few-shot, zero-shotで問題を解かせている。
- ここでもパラメタ数が大きいモデルの精度が高い。
- 最も大きいモデルでは、2桁の足し算は100%の精度で実行可能。
- しかし、その他のタスクは必ずしも高精度とはいえない。



**Figure 3.10:** Results on all 10 arithmetic tasks in the few-shot settings for models of different sizes. There is a significant jump from the second largest model (GPT-3 13B) to the largest model (GPT-3 175), with the latter being able to reliably accurate 2 digit arithmetic, usually accurate 3 digit arithmetic, and correct answers a significant fraction of the time on 4-5 digit arithmetic, 2 digit multiplication, and compound operations. Results for one-shot and zero-shot are shown in the appendix.

# 足し算（初等算術）が難しい理由

- 初等算術が難しい理由として、定性的には以下のものが考えられる。
  - マルチステップの演算を要する。
  - 厳密な処理の適用が求められる。
  - データが無限に存在する。
- 一方で、人間は適切に学習することで、これらの処理をできるようになる。
- 特に、非常に大きな数、例えば100桁の数字に対する計算も遂行できる。

$$30214528105242105108521095210952 + 950211190561895206696906822814 = ?$$

$$582109610649068213122490046840 \times 6741671349368490896390860926 = ?$$

# [ref] Mathematics Dataset

- この分野でよく用いられるデータセットの1つ。
- 右下に示したような、自然言語と数式で記述されるQuestion, Answerのペアからなるデータセット。
- さまざまな算術課題のModuleを生成するジェネレータを提供。
- pre-generated datasetとしてtrain-easy, train-medium, train-hard, interpolation, extrapolationの5つのデータセットも提供。
- 各moduleごとに、2Mのpairを用意。
- 基本的なアーキテクチャ (LSTM/RMC, Attentional LSTM/RMC, Transformer)をベースラインとして用意。

## Model Accuracy

	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
<b>Transformer</b>	30M	<b>0.76</b>	<b>0.50</b>

Figure 3: Model accuracy (probability of correct answer) averaged across modules. RMC is the relational recurrent neural network model.

## Modules

- **algebra** (linear equations, polynomial roots, sequences)
- **arithmetic** (pairwise operations and mixed expressions, surds)
- **calculus** (differentiation)
- **comparison** (closest numbers, pairwise comparisons, sorting)
- **measurement** (conversion, working with time)
- **numbers** (base conversion, remainders, common divisors and multiples, primality, place value, rounding numbers)
- **polynomials** (addition, simplification, composition, evaluating, expansion)
- **probability** (sampling without replacement)

## Examples

Question: Solve  $-42*r + 27*c = -1167$  and  $130*r + 4*c = 372$  for  $r$ .

Answer: 4

Question: Calculate  $-841880142.544 + 411127$ .

Answer:  $-841469015.544$

Question: Let  $x(g) = 9*g + 1$ . Let  $q(c) = 2*c + 1$ . Let  $f(i) = 3*i - 39$ . Let  $w(j) = q(x(j))$ . Calculate  $f(w(a))$ .

Answer:  $54*a - 30$

Question: Let  $e(l) = l - 6$ . Is 2 a factor of both  $e(9)$  and 2?

Answer: False

Question: Let  $u(n) = -n**3 - n**2$ . Let  $e(c) = -2*c**3 + c$ . Let  $l(j) = -118*e(j) + 54*u(j)$ . What is the derivative of  $l(a)$ ?

Answer:  $546*a**2 - 108*a - 118$

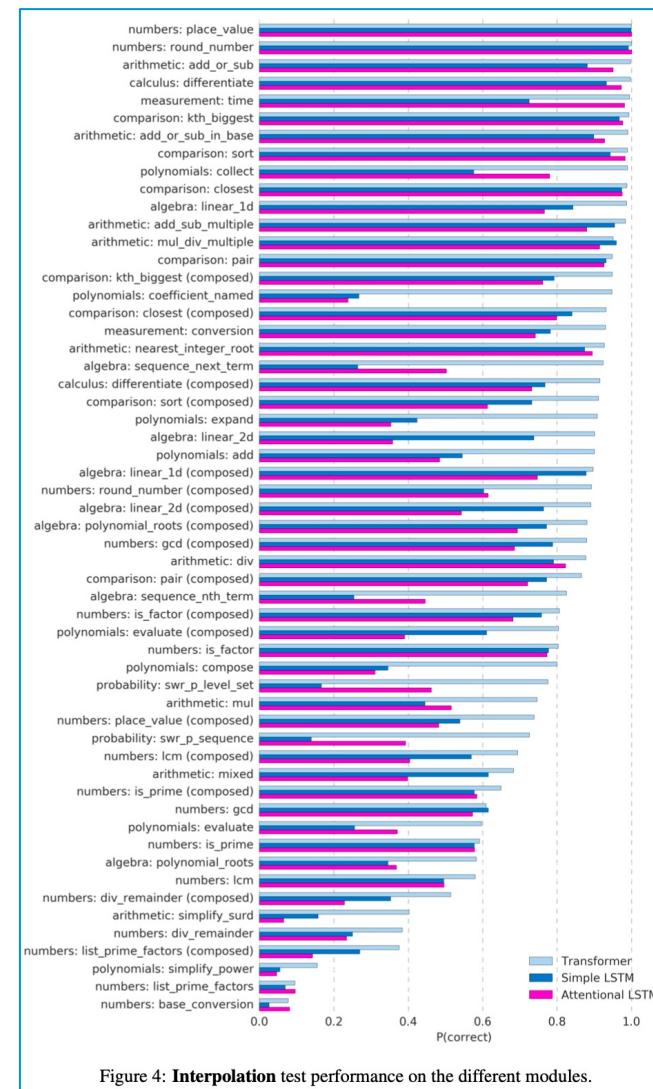
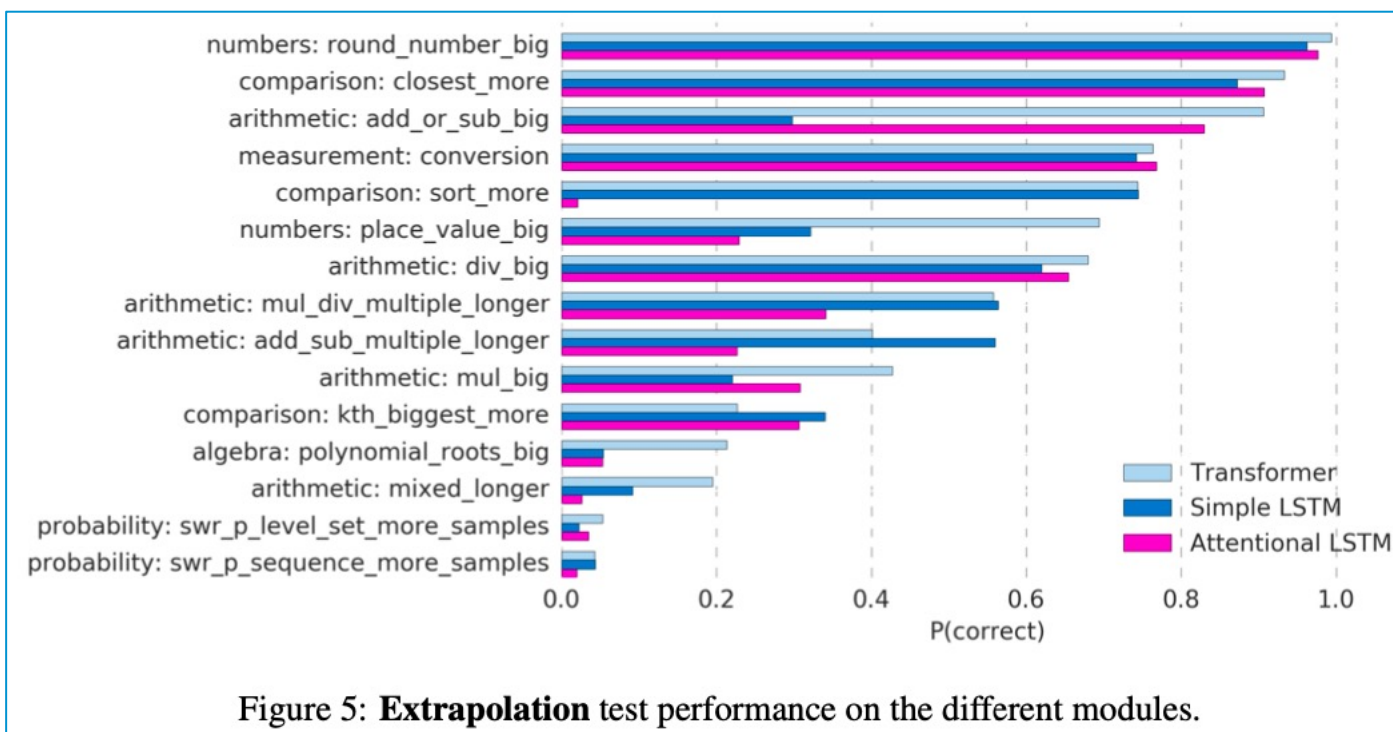
Question: Three letters picked without replacement from qqkklkqkqk. Give prob of sequence qql.

Answer: 1/110



# [ref] Mathematics Dataset

- 両図は各モデルの各moduleにおける精度を示したものの。
- 右図がinterpolationに対して、左図がextrapolationに対しての精度。
- この論文ではInterpolation, Extrapolationを数字の大きさや、長さなど、様々な軸について用意をしてはいる。しかし、多様性をもたせている分、その定義は必ずしも明確ではないとも言える。



- 前掲のMathematics datasetのように一通りの初等算術データセットは既に提案されている。
- 我々の目的からすると、一旦は四則演算など、非常に単純なものでよい。
- 以降の議論では特に、足し算を念頭に置いている。

ちなみに

- 定理証明や微積分などの高等数学への拡張など、問題をより難化させる方向の研究がある。
- それらの難化した数学課題に対するモデルの研究もされている。

ここではスコープ外。

難しい問題を解くこと、それ自体を目標としているわけではない。（詳細は後述）

# 算術課題に取り組むことは何故重要か？

- モデルとデータの大規模化だけでは達成できそうにないこと  
既に示した通りGPT-3でも、PaLMでも解けているわけではない。
- 具体的な課題を用意することで、問題がクリアになると期待できること  
汎用人工知能を作りたいと思っても、具体的な課題なしでは抽象的な議論に留まりがち。
- 深層学習で、この手の問題を解決できないと思われる特別な理由はないこと  
Connectionismを思えば、ニューラルネットで解けてもよいのではないか。
- 画像認識など多くのタスクで、論理的処理の援用は有用であること  
算術に限らない範囲で機能する良い帰納バイアスを見つけられれば、さまざまなタスクで説明可能性の向上が期待できる

新たなアーキテクチャ開発の手がかりとなるのではないか

# 足し算は本当に解けないのか？

ところで、足し算は本当に解けないのか？

そんなことはない。例えば、四則演算を実行するプログラムを直接実装すればよい。

ニューラルネットを用いるアプローチにおいてもさまざまな研究が行われている。

- 大規模データ&モデルのアプローチでMathematics datasetのExtrapolationにおいて、ある程度高い精度は達成可能。 [arXiv:2010.14701]
- 計算過程を直接、教師ラベルとして与えることで、汎化性のあるモデルが得られる。 [arXiv:2112.00114]
- 複数ステップで解く問題を強化学習の課題とみなす。環境を定義し、階層型強化学習とカリキュラム学習でモデルを訓練することで、汎化性のあるモデルが得られる。 [arXiv:1809.08590]
- T5(Transformer)でも、入力の表現を工夫することで、60桁程度までの足し算・引き算は行えるようになった。50桁程度の数字で訓練することで、60桁までの外挿もある程度は可能になった。 [arXiv:2102.13019]

しかし、これらの多くの研究では、ドメイン特化のモジュールが導入されている。

つまり開発者の知識によって、タスクを部分的に解いているともいえる。

人間がドメインに応じて対応するのではなく、種々のタスクが与えられたときに、データから学ぶなどして、自律的に解決するようなシステムが望ましい。

# そもそも何ができたら足し算ができたと言えるのか？

1. 大量のデータで、データを与えた範囲に限って解くことができる。数%の間違いは許容する。
2. 大量のデータで、データを与えた範囲に限って解くことができる。数%の間違いも許容しない。
3. 少量のデータで、データを与えた範囲に限って解くことができる。
4. 大量のデータで、データを与えた範囲を超えて解くことができる。数%の間違いは許容する。
5. 大量のデータで、データを与えた範囲を超えて解くことができる。数%の間違いは許容しない。
6. 大量のデータで、データを与えた範囲を超えて解くことができる。理屈のつく間違いのみ許容する。
7. 少量のデータで、データを与えた範囲を超えて解くことができる。理屈のつく間違いのみ許容する。
8. . . .
9. 人間にとって馴染みのある方法で解くことができる。
10. 課題解決の過程を閲覧できる。
11. 課題解決の過程をシステムが説明できる。
12. 数の大小の理解や、数の概念を形成し、自分が何を行っているかを認識しながら解くことができる。
13. . . .

1から5(部分的に6,7)は、汎化性能の問題、つまり外挿領域で高い精度で予測できるかという問題といえる。

9から12(部分的に6, 7)は、説明可能性を備えているかどうかを重視している。

- 指摘1：いや、算術課題程度は、実はもう解けているでしょう。
- 応答1：Yes and No. 見方によっては解けているが、解けていないとも捉えられる。

- 指摘1：いや、算術課題程度は、実はもう解けているでしょう。
- 応答1：Yes and No. 見方によっては解けているが、解けていないとも捉えられる。

内挿領域では解けていると思える。

データをどんどん増やして、実際に現れる範囲のデータを十分尽くすことができれば、解けているとも思える。

- 指摘1：いや、算術課題程度は、実はもう解けているでしょう。
- 応答1：Yes and No. 見方によっては解けているが、解けていないとも捉えられる。  
外挿と開発者の知識について考察する必要がある。  
外挿について考えていると、そもそも解けない問題設定のようにも思えてくる。  
開発者の知識について考えていると、算術課題それ自体を解きたいわけではないことに気づく。



- 指摘1：いや、算術課題程度は、実はもう解けているでしょう。
- 応答1：Yes and No. 見方によっては解けているが、解けていないとも捉えられる。  
外挿と開発者の知識について考察する必要がある。  
外挿について考えていると、そもそも解けない問題設定のようにも思えてくる。  
開発者の知識について考えていると、算術課題それ自体を解きたいわけではないことに気づく。

- 指摘1：いや、算術課題程度は、実はもう解けているでしょう。
- 応答1：Yes and No. 見方によっては解けているが、解けていないとも捉えられる。  
外挿と開発者の知識について考察する必要がある。  
外挿について考えていると、そもそも解けない問題設定のようにも思えてくる。  
開発者の知識について考えていると、算術課題それ自体を解きたいわけではないことに気づく。

有限個のデータで、無限個のテストデータにおいて高い精度で予測するのは原理的にできないのでは？  
そもそも、訓練データ外でのデータの振る舞いは規定されていないのでは。  
有限個の入出力ペアだけから、すべての整数に対する足し算の関係は普通は定まらない。

人間は何故できるのか。具体例の丸暗記ではなく、自然言語などによって足し算を規定している。  
それらのルールを適用することで、見たことのない大きな数字についても”正しく”計算できる。

- 指摘1：いや、算術課題程度は、実はもう解けているでしょう。
- 応答1：Yes and No. 見方によっては解けているが、解けていないとも捉えられる。  
外挿と開発者の知識について考察する必要がある。  
外挿について考えていると、そもそも解けない問題設定のようにも思えてくる。  
開発者の知識について考えていると、算術課題それ自体を解きたいわけではないことに気づく。

有限個のデータで、無限個のテストデータにおいて高い精度で予測するのは原理的にできないのでは？  
そもそも、訓練データ外でのデータの振る舞いは規定されていないのでは。  
有限個の入出力ペアだけから、すべての整数に対する足し算の関係は普通は定まらない。

人間は何故できるのか。具体例の丸暗記ではなく、自然言語などによって足し算を規定している。  
それらのルールを適用することで、見たことのない大きな数字についても”正しく”計算できる。

- 指摘2：人間は必ずしも入出力のペアだけから解いているわけではないだろう。人間でも、その問題設定なら外挿はできないのではないか。
- 応答2：そうかもしれない。そうだとすると、機械学習のタスクの枠組みを変更する必要があるかもしれない。

- 指摘1：いや、算術課題程度は、実はもう解けているでしょう。
- 応答1：Yes and No. 見方によっては解けているが、解けていないとも捉えられる。  
外挿と開発者の知識について考察する必要がある。  
外挿について考えていると、そもそも解けない問題設定のようにも思えてくる。  
開発者の知識について考えていると、算術課題それ自体を解きたいわけではないことに気づく。

有限個のデータで、無限個のテストデータにおいて高い精度で予測するのは原理的にできないのでは？  
そもそも、訓練データ外でのデータの振る舞いは規定されていないのでは。  
有限個の入出力ペアだけから、すべての整数に対する足し算の関係は普通は定まらない。

人間は何故できるのか。具体例の丸暗記ではなく、自然言語などによって足し算を規定している。  
それらのルールを適用することで、見たことのない大きな数字についても”正しく”計算できる。

一方で、そこには、アルゴリズム記述長が短いなどの、“自然なルール”というものもありそう。  
足し算程度の複雑さであれば、自然なものが見つかってくれてもよさそうな気もする。  
---> 足し算を解くというのは「足し算を解くためのアルゴリズムを見つけて、それを実行すること」とみなせる。

- 指摘1：いや、算術課題程度は、実はもう解けているでしょう。
- 応答1：Yes and No. 見方によっては解けているが、解けていないとも捉えられる。  
外挿と開発者の知識について考察する必要がある。  
外挿について考えていると、そもそも解けない問題設定のようにも思えてくる。  
開発者の知識について考えていると、算術課題それ自体を解きたいわけではないことに気づく。

- 指摘1：いや、算術課題程度は、実はもう解けているでしょう。
- 応答1：Yes and No. 見方によっては解けているが、解けていないとも捉えられる。  
外挿と開発者の知識について考察する必要がある。  
外挿について考えていると、そもそも解けない問題設定のようにも思えてくる。  
開発者の知識について考えていると、算術課題それ自体を解きたいわけではないことに気づく。

算術課題特化のモジュールを開発者が組み込んで解けたとしても、その方法は他の問題には適用できない。  
さまざまあるタスクに対して、開発者がその都度ドメインごとに対応する必要があるものは、望ましくはない。

しかし、開発者がタスクのための知識を全く導入しないのであれば、システムは何の仮定もないところからタスクを解くためのアルゴリズムを見つけなくてはならない。

帰納バイアスはアルゴリズムの空間における探索空間を狭めてくれる。  
一方で、ドメインに特化しすぎた帰納バイアスを入れると、多くのタスクでは機能しなくなる。  
帰納バイアスとドメイン特化はトレードオフの関係にある。

- 指摘1：いや、算術課題程度は、実はもう解けているでしょう。
- 応答1：Yes and No. 見方によっては解けているが、解けていないとも捉えられる。  
外挿と開発者の知識について考察する必要がある。  
外挿について考えていると、そもそも解けない問題設定のようにも思えてくる。  
開発者の知識について考えていると、算術課題それ自体を解きたいわけではないことに気づく。

算術課題特化のモジュールを開発者が組み込んで解けたとしても、その方法は他の問題には適用できない。  
さまざまあるタスクに対して、開発者がその都度ドメインごとに対応する必要があるものは、望ましくはない。

しかし、開発者がタスクのための知識を全く導入しないのであれば、システムは何の仮定もないところからタスクを解くためのアルゴリズムを見つけなくてはならない。

帰納バイアスはアルゴリズムの空間における探索空間を狭めてくれる。  
一方で、ドメインに特化しすぎた帰納バイアスを入れると、多くのタスクでは機能しなくなる。  
帰納バイアスとドメイン特化はトレードオフの関係にある。

- 指摘3：では、**どんな帰納バイアスが求められているのか？**
- 応答3：これはもしかすると**行動を自分で定義できる**ということなのではないか？

# どんな方法で解けるだろうか？

- 訓練データを仮想的に分割する？  
それぞれのデータのペアを対応付けするだけでは、同じ問題に陥るだろう。  
データを関連づける枠組みが必要そう。分割されたデータで訓練ができればよい。  
しかし、これは一見不可能な枠組みにも思えるか。
- 機械学習のタスクの枠組みを変える。  
入出力のペアだけからでは解けない。少なくとも人間はもっとリッチな情報を使っている。  
例えば、タスクの記述を自然言語で与えられる。  
であれば、機械学習の問題設定もそのような拡張が必要？
- 行動のようなものを自分で定義できる？  
何故これが必要か。環境を人手で与えてしまうと、それがドメイン固有の解決策になっている。  
そうではなくて、その問題にふさわしい行動をシステム自らが定義できればよい。