Autoformalization with Large Language Models-hoho

论文试图解决什么问题？

大规模语言模型在形式化语言生成时取得不错的成绩（代码生成），但在形式化数学证明上的应用却很少。本文试图验证现有的大规模语言模型是否在形式化数学证明上也有不错的表现。

这是否是一个新的问题？

这不是一个新问题，但本文的模型达到一个新的SOTA

这篇文章要验证一个什么科学假设？

验证现有的大规模语言模型在形式化数学证明上也有不错的表现。

有哪些相关研究？如何归类？谁是这一课题在领域内值得关注的研究员？

当时的研究工作可以归为两类：

1. 基础模型算法，包括基于监督学习与无监督学习的方法。本文使用的自监督的方法。
2. 解决训练数据稀缺的问题：使用强化学习的方法来减少人工的过多干预

比较值得关注的研究员有Ilya Sutskever，参与Codex的研究，相关论文《Evaluating Large Language Models Trained on Code》，他还有著有跟文本类似的研究的《**Generative Language Modeling for Automated Theorem Proving**》

论文中提到的解决方案之关键是什么？

可以将数学自然语言描述生成代码看作是一个模型翻译过程。

为了验证大型语言模型生成的数学代码是有效的，使用了一种self-imporvement cycle操作（expert iteration）：

1. 首先有一个基础的证明器M_0（neural theorem prover，其根本也是一个语言模型），以及以及正确形式化好的数学题代码数据集A（大型语言模型生成的）
2. 迭代$i = 1...N$：使用M_{i-1}证明器验证数据集A，将验证成功的样本S_i跟所有的数据拼接成新的数据集$A_i = \cup_{j \leq i} S_i \cup B$ ，继续训练，获得一个新的证明器M_i

（hoho_todo: 1. 证明器neural theorem prover如何运作的？ 2. 关于数据拼接那不就当前的正确数据不断double?）

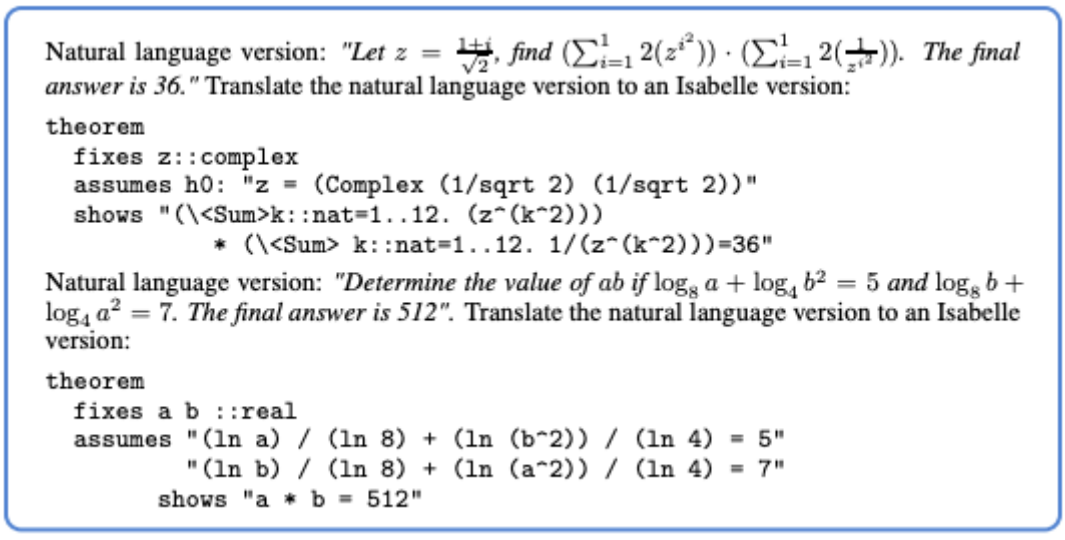
论文中的实验是如何设计的？

1. 由于数据集中的数学题目多以问题形式出现而非命题，故预处理为“题目描述+题目答案”的形式：

   $Problem_Statement The final answer is $Answer.

2. 将数学命题用Codex或PaLM生成为代码，使用prompt的技巧，将两到数学题描述形成prompt,格式为：

   Natural language version: $Natural_Language_Statement.
   Translate the natural language version to an Isabelle version:



Natural language version: *"Let* $z = \frac{1+i}{\sqrt{2}}$*, find* $(\sum_{i=1}^{1} 2(z^{i^2})) \cdot (\sum_{i=1}^{1} 2(\frac{1}{z^{i^2}}))$*. The final answer is 36."* Translate the natural language version to an Isabelle version:

```
theorem
  fixes z::complex
  assumes h0: "z = (Complex (1/sqrt 2) (1/sqrt 2))"
  shows "(\<Sum>k::nat=1..12. (z^(k^2)))
           * (\<Sum> k::nat=1..12. 1/(z^(k^2)))=36"
```

Natural language version: *"Determine the value of ab if* $\log_8 a + \log_4 b^2 = 5$ *and* $\log_8 b + \log_4 a^2 = 7$*. The final answer is 512".* Translate the natural language version to an Isabelle version:

```
theorem
  fixes a b ::real
  assumes "(ln a) / (ln 8) + (ln (b^2)) / (ln 4) = 5"
          "(ln b) / (ln 8) + (ln (a^2)) / (ln 4) = 7"
        shows "a * b = 512"
```

   并且在解码时使用贪心解码（greedy decoding）

3. 使用dataset中的人类ground truth计算BLEU
4. 验证模型输出的错误用例，分析譬如成功率

5. 对生成的结果形成新的数据集，然后进行expert iteration（见“论文中提到的解决方案之关键是什么？”一节）

用于定量评估的数据集是什么？代码有没有开源？

- MATH数据集：https://github.com/hendrycks/math/
- MiniF2F: https://github.com/openai/miniF2F

没开源代码

论文中的实验及结果有没有很好地支持需要验证的科学假设？

实验证明，大规模语言模型可以很好的将数学题描述转换为代码实现，而且模型规模越大，效果越好（Codex比PaLM更优，可能由于Codex预训练了更多的相关数据）

进行expert iteration后，本文中的neural theorem证明性能优于当前的5.6%

这篇论文到底有什么贡献？

验证了大规模语言模型可以将数学自然语言描述转化为形式化表示，并且证明了用这些训练生成的形式化数学表达可以提升neural theorem prover的可行性。

下一步呢？有什么工作可以继续深入？

1. 因为本文用的是静态模型，当需要形式化更大的数学问题时可能需要添加更多的新数学标记，保持足够大上下文信息，于是难以泛化到新的形式化数学问题的一小部分上，可以考虑一些持续训练、expert iteration、cycle-consistency-based training或者in-context学习的新应用
2. 在生成更大形式化数学语言时，Retrieval-augmented 语言模型譬如memorizing transformer可以解决长序列的召回问题。