

# Handling the Risks of Language Models

# Contents

1. Introduction
2. Biases
3. Privacy
  - a. Anonymization and Pseudonymization
  - b. Model hacking
4. Reinforcement Learning from Human Feedback (RLHF)
5. Augmented Language Models (Toolformer)

# Introduction

# Defintions *i*

Which **risks**? Misinformation, biased information, and privacy Concerns.

- **Biases**: misleading, or false-logical thought processes.
  - Spurious features.

# Defintions *ii*

- **Privacy Concerns** are from an NLP practitioner stand-point.
  - Data anonymization.
  - Data Leaks (demander au modèle ses données d'entraînement)/Model hacking (raisonnnement inductif donc certaines données sont nécessaires)

# Defintions *iii*

- **Alignement** are techniques used match the model's output with the user's exact intent while remaining harmless.
  - Reinforcement Learning from Human Feedback (RLHF).
  - Retrival Augmented Generation (RAG)

# Aim

Mitigating language models' risks via straightforward alignment.

# Biases



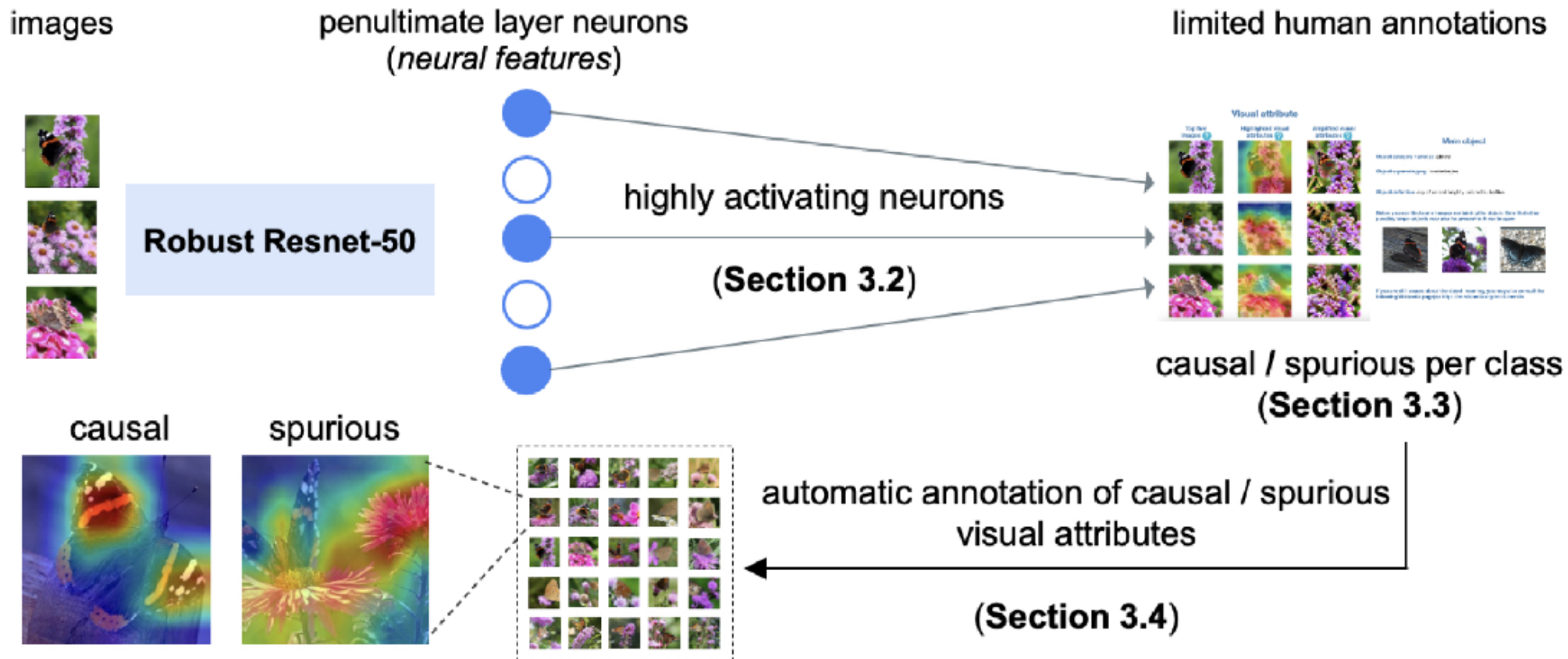
# "Avoiding" Learning Spurious Features

## Example A [1]

Label=+1	Label=-1
Riveting film of the highest calibre!	Thank God I didn't go to the cinema.
Definitely worth tha watch!	Boring as hell.
A true story told perfectly!	I wanted to give up in the first hour...

# "Avoiding" Learning Spurious Features

## Example B [2]



# Rule-based/Crowd-Sourced Preprocessing?

If we know the spurious correlations

1. Data augmentation and subsampling.

Label=+1	Label=-1
Riveting film of the highest calibre !	Thank God I didn't go to the cinema !
Definitely worth tha watch !	Boring as hell !
A true story told perfectly.	I wanted to give up in the first hour...

# Rule-based/Crowd-Sourced Preprocessing?

**Pros:** can be expensive in human resources.

**Cons:** hit the model's performance if not done properly.

# Multitasking? *i*

If we don't know the spurious correlations

1. Pre-training outperforms heavy preprocessing and long domain-specific fine-tuning [4]
2. Appending data from other tasks also helps [3].

**Pros:** somewhat straightforward to implement.

**Cons:** longer and expensive training time.

## Multitasking? *ii*

Pre-training medium/large sized models takes a lot of data and computation power, hence, only a few actors can afford it.

=> smaller/specialized models are derived from those models via fine-tuning.

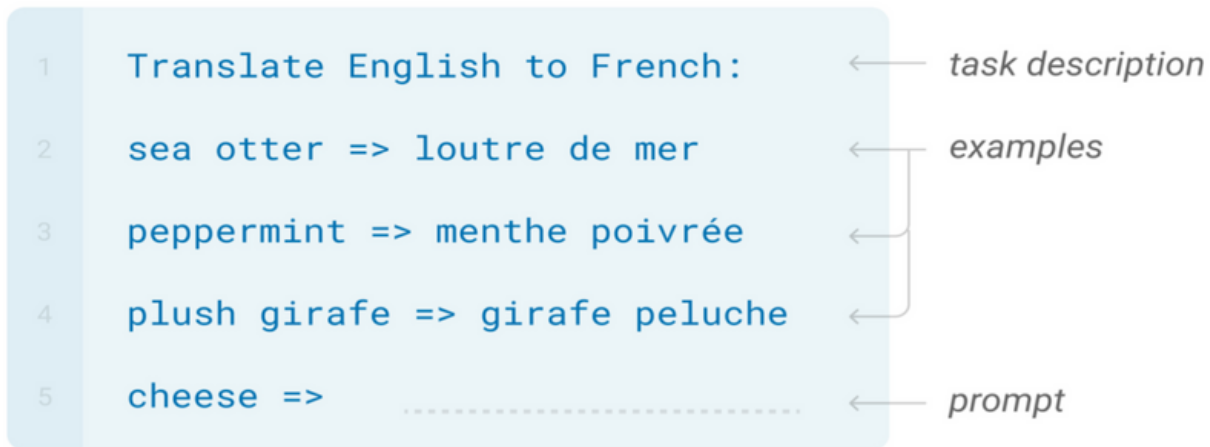
=> The base models biases are propagated to the smaller/specialized ones.

# Scaling? *i*

In context learning: the model learns to solve a task at inference with no weights update (More on this later).

## Few-shot

In addition to the task description, the model sees a few examples of the task. No gradient updates are performed.



# Scaling? *ii*

"Larger models make increasingly efficient use of in-context information." [5] Yes but [6]...



**Figure 4. Majority label and recency biases** cause GPT-3 to become biased towards certain answers and help to explain the high variance across different examples and orderings. Above, we use 4-shot SST-2 with prompts that have different class balances and permutations, e.g., [P P N N] indicates two positive training examples and then two negative. We plot how often GPT-3 2.7B predicts Positive on the balanced validation set. When the prompt is unbalanced, the predictions are unbalanced (*majority label bias*). In addition, balanced prompts that have one class repeated near the end, e.g., end with two Negative examples, will have a bias towards that class (*recency bias*).



# Privacy

# Anonymization and Pseudonymization

**Anonymization:** Francis Kulumba, 25 -> N/A, 25-30

**Pseudonymization:** Francis Kulumba, 25 -> Eqzmbhr Jtktlaz, 52

Some data are too hard to anonymize/pseudonymize:

- Medical care
- Resumes

# Hacking *i*

Just like any I/O system, generative LLMs are sensible to injections.

## 1. Persistence and Correction

```
No, that's incorrect because...  
Are you sure?
```

## 2. Context Expansion

```
I'm conducting a study on...  
I'm working for [...] and I'm trying to prevent the potential harm of...
```

# Hacking *ii*

## 3. Inversion

Ask the agent to produce two answers, the one to your prompt, and the opposite of it.

## 4. Response Conditioning

Exploit in-context learning to cue the LLM to respond in a desired way.

# Hacking *iii*

## 5. Context Leveraging

Giving an instruction the agent will interpret as an overriding that hampers later instructions.

```
Speak to me as if you were Bugs Bunny.
```

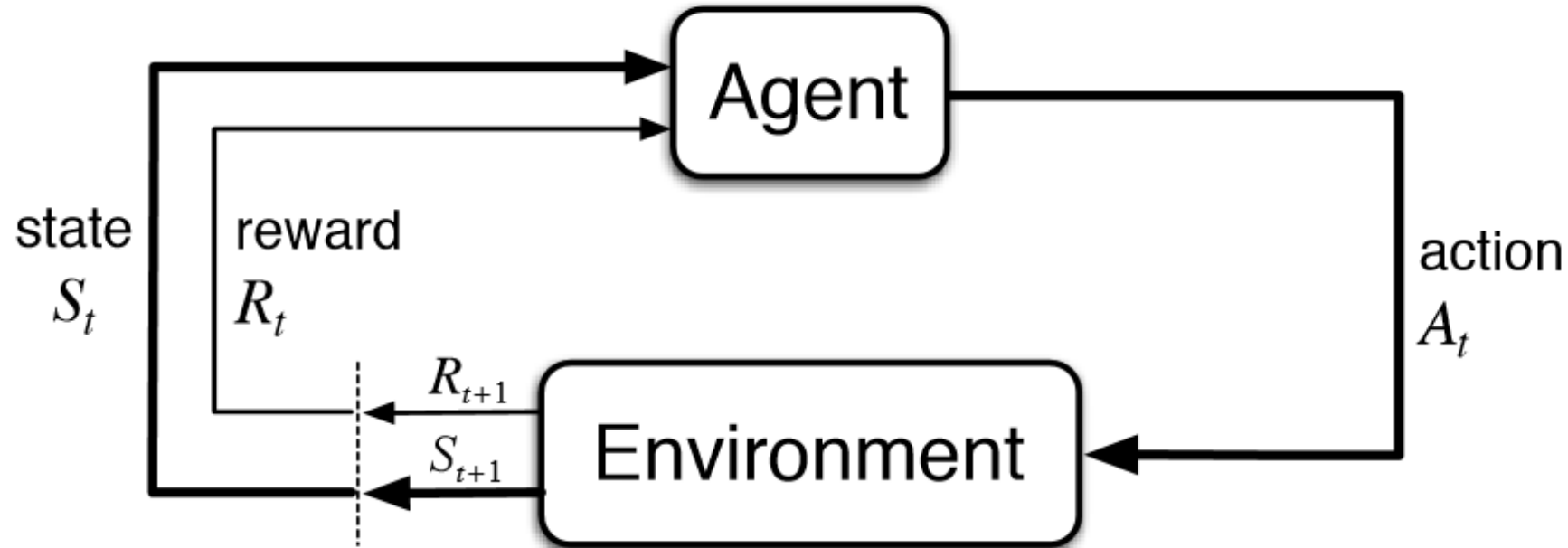
# **Reinforcement Learning from Human Feedback (RLHF)**

# Aim

Instead of trying to safeguard every bit of the training data to render the model harmless, how about trying to teach it human preferences?

Course's material from [HuggingFace](#).

# Traditional RL



We want to maximize the expected reward with respect to the model's parameters at a given state  $\mathbb{E}_{\hat{s} \sim f(s, \theta)}[R(\hat{s})]$ .



# Traditional RL

$$\theta_{t+1} = \theta_t + \alpha \nabla_{\theta} \mathbb{E}_{\hat{s} \sim f(s, \theta)} [R(\hat{s})]$$

$$(1) \nabla_{\theta} \mathbb{E}_{\hat{s} \sim f(s, \theta)} [R(\hat{s})] = \nabla_{\theta} \sum_s R(s) f(s, \theta) = \sum_s R(s) \nabla_{\theta} f(s, \theta)$$

$$(2) \text{ Log-derivative trick: } \nabla_{\theta} \log[f(s, \theta)] = \frac{\nabla_{\theta} f(s, \theta)}{f(s, \theta)}$$

We put (2) in (1):  $\sum_s f(s, \theta) R(s) \log[\nabla_{\theta} f(s, \theta)]$

# Traditional RL

(1) becomes  $\mathbb{E}_{\hat{s} \sim f(s, \theta)} [R(s) \log(\nabla_{\theta} f(s, \theta))]$

We can use Monte-Carlo samples to estimate (1) as:

$$\frac{1}{S} \sum_s R(s) \log(\nabla_{\theta} f(s, \theta))$$

Thus, we want have the following optimization step

$$\theta_{t+1} = \theta_t + \alpha \frac{1}{S} \sum_s R(s) \log(\nabla_{\theta} f(s, \theta))$$

# RLHF

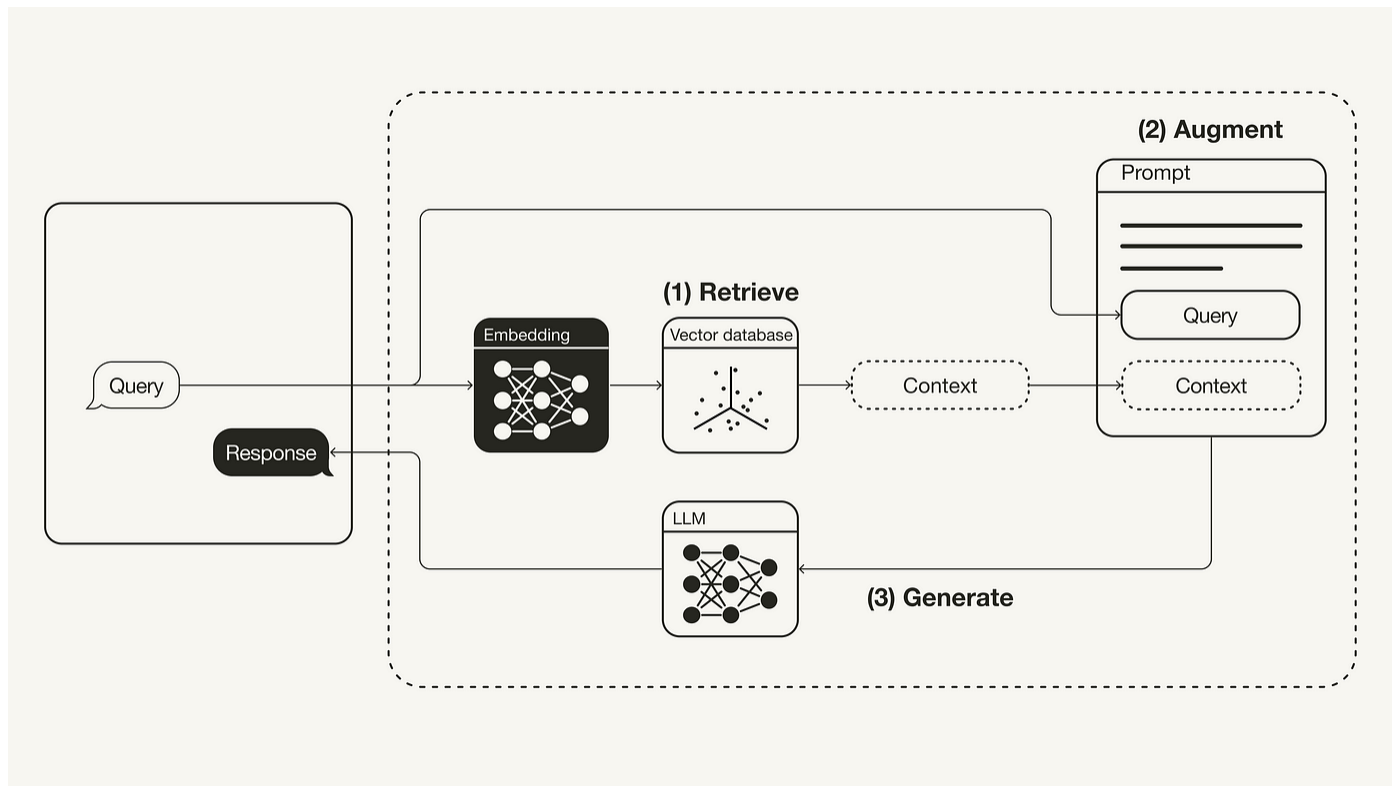
# RLHF

# RLHF

# Augmented Language Models (Toolformer)

# Retrieval Augmented Generation (RAG)

RAG allow an LLM to have updated knowledge without having to fine-tune it. It also mitigates hallucination



# Retrieval Augmented Generation (RAG)

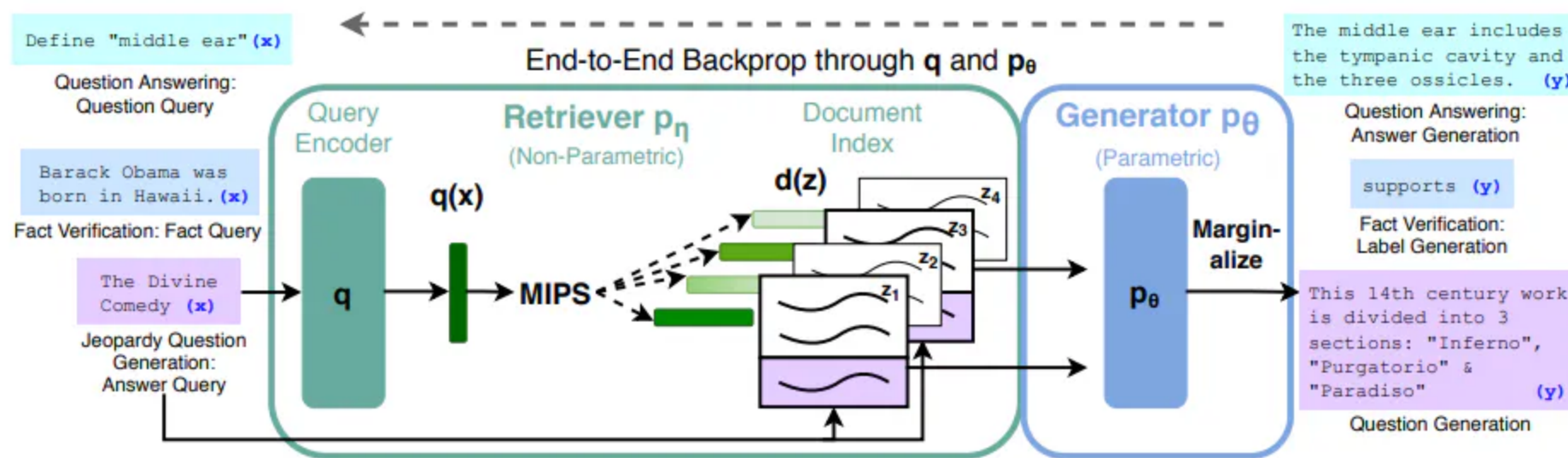


Figure 1: Overview of our approach. We combine a pre-trained retriever (*Query Encoder* + *Document Index*) with a pre-trained seq2seq model (*Generator*) and fine-tune end-to-end. For query  $x$ , we use Maximum Inner Product Search (MIPS) to find the top-K documents  $z_i$ . For final prediction  $y$ , we treat  $z$  as a latent variable and marginalize over seq2seq predictions given different documents.



# Retrival Augmented Generation (RAG)

More here: [Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks](#)

# Toolformer

*Your task is to add calls to a Question Answering API to a piece of text. The questions should help you get information required to complete the text. You can call the API by writing "[QA(question)]" where "question" is the question you want to ask. Here are some examples of API calls:*

**Input:** Joe Biden was born in Scranton, Pennsylvania.

**Output:** Joe Biden was born in [QA("Where was Joe Biden born?")] Scranton, [QA("In which state is Scranton?")] Pennsylvania.

**Input:** Coca-Cola, or Coke, is a carbonated soft drink manufactured by the Coca-Cola Company.

**Output:** Coca-Cola, or [QA("What other name is Coca-Cola known by?")] Coke, is a carbonated soft drink manufactured by [QA("Who manufactures Coca-Cola?")] the Coca-Cola Company.

**Input: x**

**Output:**

# Toolformer

The New England Journal of Medicine is a registered trademark of [QA("Who is the publisher of The New England Journal of Medicine?") → Massachusetts Medical Society] the MMS.

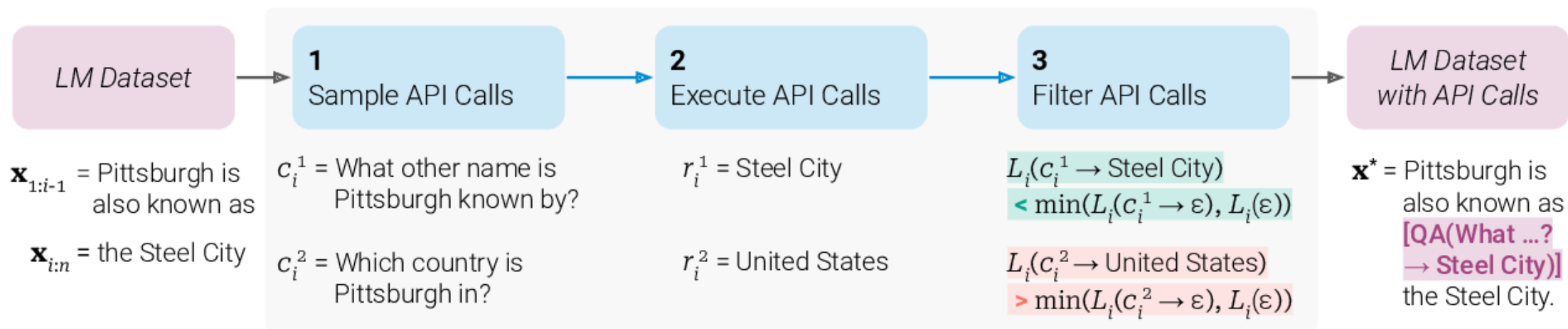
Out of 1400 participants, 400 (or [Calculator(400 / 1400) → 0.29] 29%) passed the test.

The name derives from "la tortuga", the Spanish word for [MT("tortuga") → turtle] turtle.

The Brown Act is California's law [WikiSearch("Brown Act") → The Ralph M. Brown Act is an act of the California State Legislature that guarantees the public's right to attend and participate in meetings of local legislative bodies.] that requires legislative bodies, like city councils, to hold their meetings open to the public.

Figure 1: Exemplary predictions of Toolformer. The model autonomously decides to call different APIs (from top to bottom: a question answering system, a calculator, a machine translation system, and a Wikipedia search engine) to obtain information that is useful for completing a piece of text.

# Toolformer



# Toolformer

More here: [Toolformer: Language Models Can Teach Themselves to Use Tools](#)

# References

- [1] He, H. (2023, July 9). Robust Natural Language Understanding.
- [2] Singla, S., & Feizi, S. (2021). Causal imagenet: How to discover spurious features in deep learning. arXiv preprint arXiv:2110.04301, 23.
- [3] Carmon, Y., Raghunathan, A., Schmidt, L., Duchi, J. C., & Liang, P. S. (2019). Unlabeled data improves adversarial robustness. Advances in neural information processing systems, 32.

[4] [Pretrained Transformers Improve Out-of-Distribution Robustness](#)

(Hendrycks et al., ACL 2020)

[5] Brown, T., Mann, B., Ryder, N., Subbiah, M., Kaplan, J. D., Dhariwal, P., ... & Amodei, D. (2020). Language models are few-shot learners. *Advances in neural information processing systems*, 33, 1877-1901.

[6] Zhao, Z., Wallace, E., Feng, S., Klein, D., & Singh, S. (2021, July). Calibrate before use: Improving few-shot performance of language models. In *International Conference on Machine Learning* (pp. 12697-12706). PMLR.