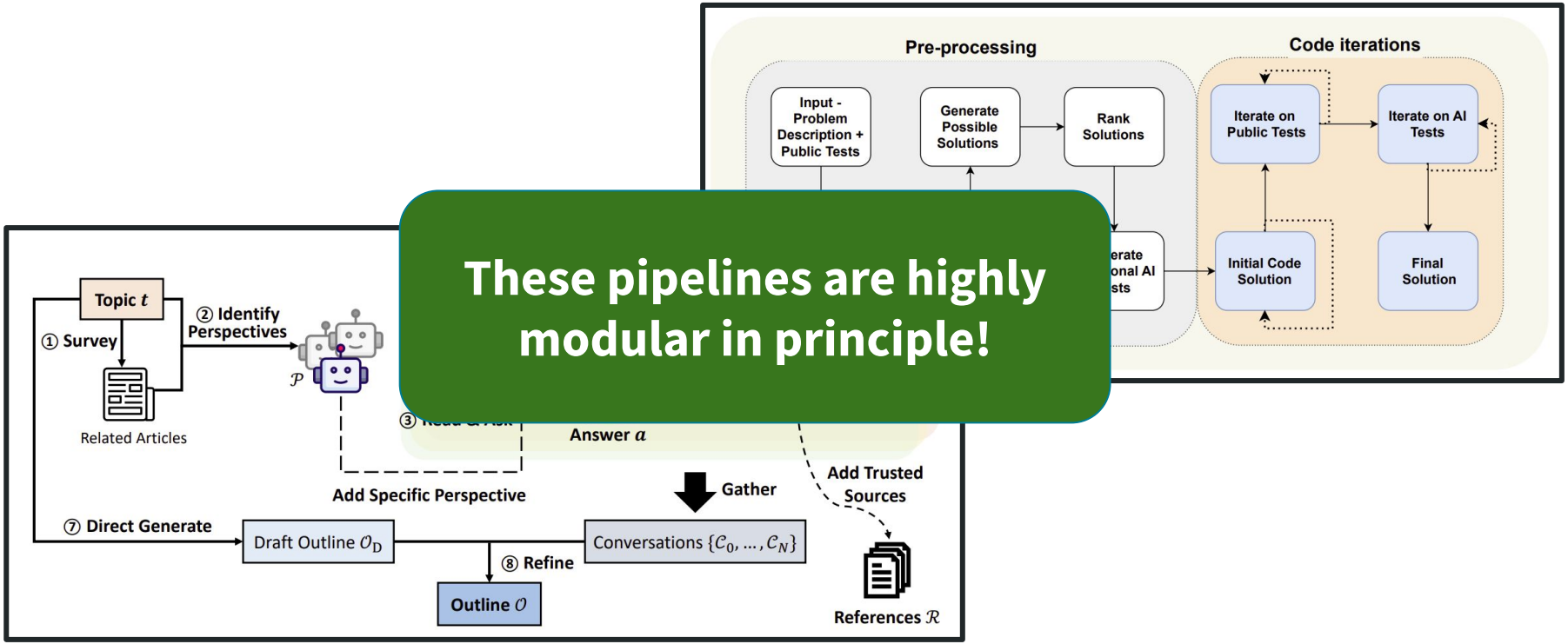
DSPy

Optimizing Instructions and Demonstrations for Multi-Stage LM Programs

Krista Opsahl-Ong*, **Michael J. Ryan***, Josh Purtell,
David Broman, Chris Potts, Matei Zaharia, Omar Khattab

LM systems with multiple modules are advancing tasks

Both academia and industry are rapidly applying these to many problems!



These pipelines are highly modular in principle!

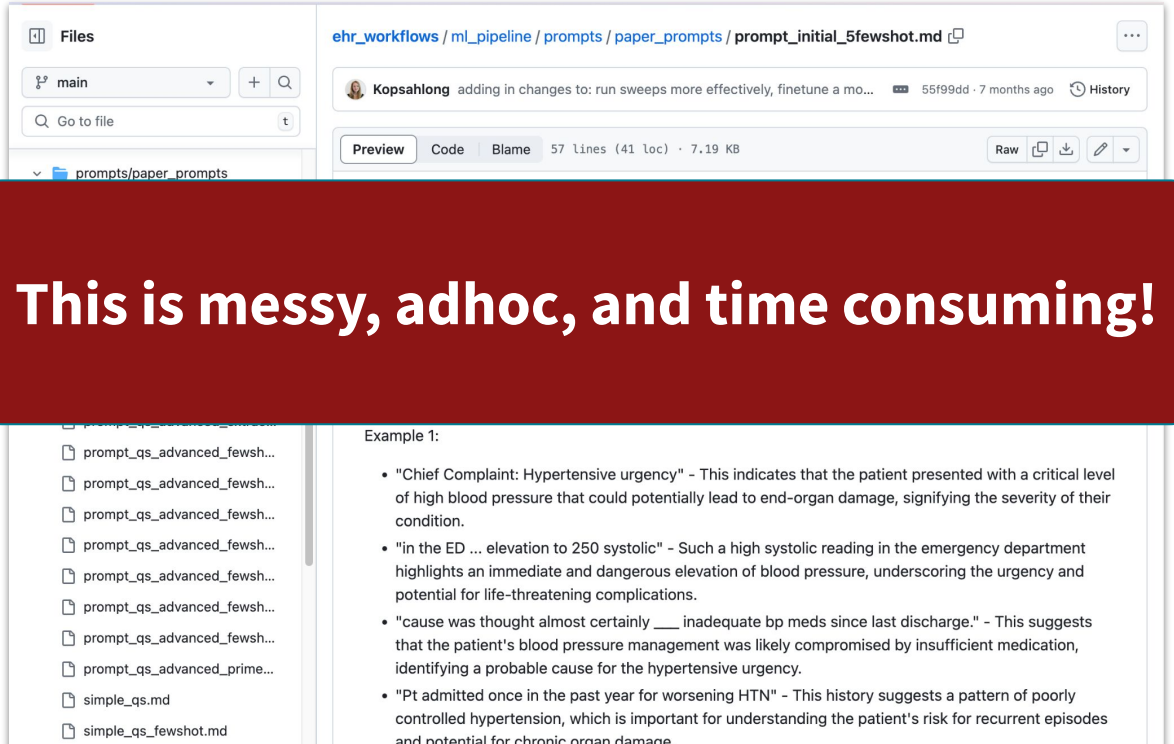
Y Shao, Y Jiang, T Kanell, P Xu, O Khattab, M Lam “Assisting in Writing Wikipedia-like Articles From Scratch with Large Language Models”

Tal Ridnik, Dedy Kredo, Itamar Friedman “Code Generation with AlphaCodium: From Prompt Engineering to Flow Engineering”

But in practice, they often involve extensive manual prompt engineering…

(a real example from an old repo of mine…)

50+ files containing different prompt variations



This is messy, adhoc, and time consuming!

How can we optimize prompts in these pipelines in a more systematic, modular way?

Optimizing Instructions and Demonstrations for Multi-Stage Language Model Programs

Krista Opsahl-Ong[1*], Michael J Ryan[1*], Josh Purtell[2],
David Broman[3], Christopher Potts[1], Matei Zaharia[4], Omar Khattab[1]

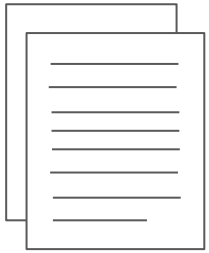
[1]Stanford University, [2]Basis, [3]KTH Royal Institute of Technology [4]UC Berkeley

Problem Setting

No access to log probs or intermediate labels!

Inputs:

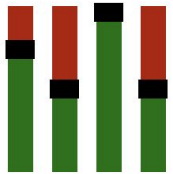
Training/ Validation Input



LM Program:

```
for i in range(2):
    query = "context, question-> search_query"
    context.append( retrieve "search_query" )
answer = "context, question-> answer"
```

Metric



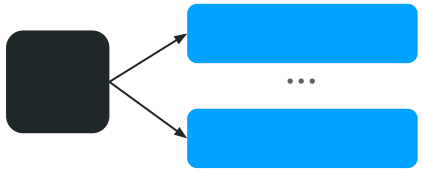
Outputs:

```
Given the question and context passages, generate the correct answer.

Question: The Victorians is a documentary series written by an author born in what year?
Context: [1] The Victorians - Their Story In Pictures is ...
         [2] Jeremy Dickson Paxman(born 11 May 1950) is an English...
Rationale: The Victorians was written by Jeremy Paxman.  Jeremy Paxman was born in 1950.
Answer: 1950

Question: Which actor played in both…
```

Instructions

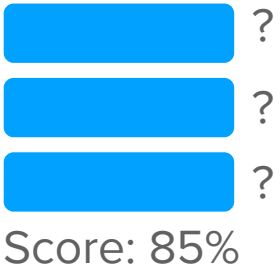
Few-Shot Ex.

Key Challenges



Prompt Proposal.

Searching over all possible strings is intractable, especially as we add in multiple modules we need to optimize. Instead, we need to propose a *small set* of *high quality* options.



Credit Assignment.

We need efficient ways of inferring how each prompt variable contributes to performance, so that we can find the best set for our program.

We propose MIPRO, which works in 3 steps:

Prompt Proposal

1. Bootstrap Task Demonstrations

2. Propose Instruction Candidates with a proposer LM

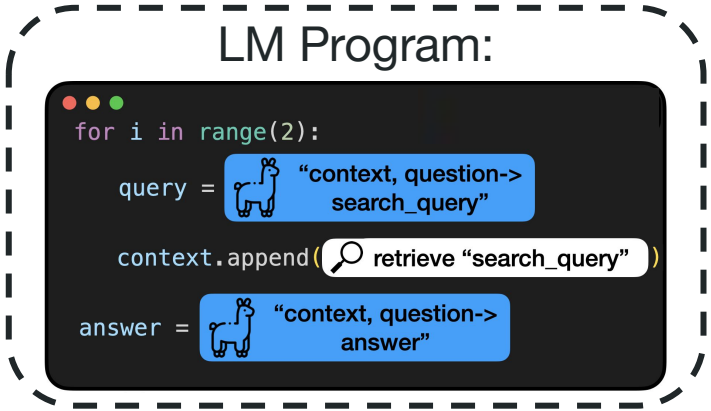
Credit Assignment

3. Compose optimize LM program w/ candidates using Bayesian Learning

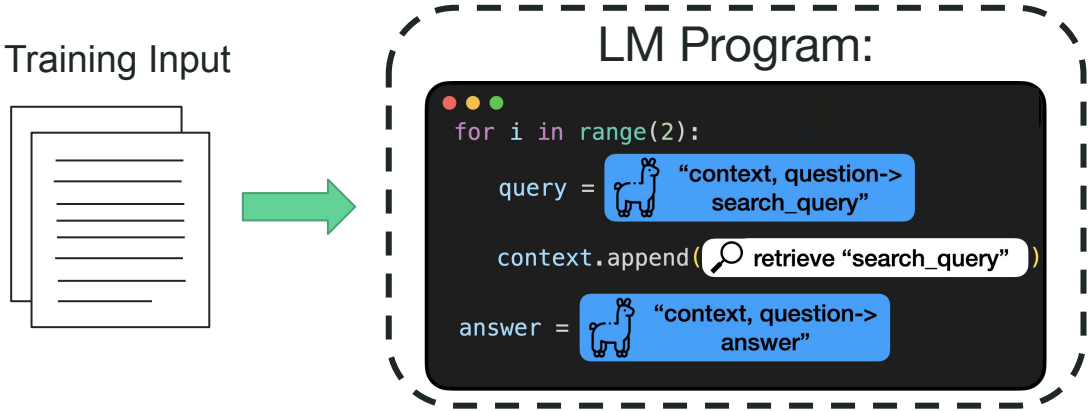
MIPRO is currently built and available in the open-source library, DSPy!

Step 1: Bootstrap Task Demonstrations

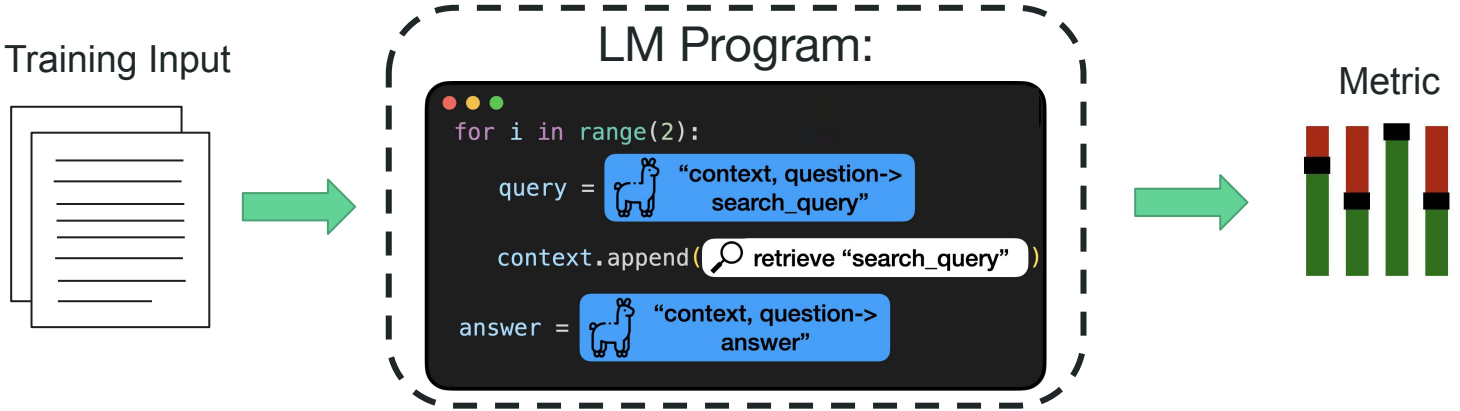
Step 1: Bootstrap Task Demonstrations



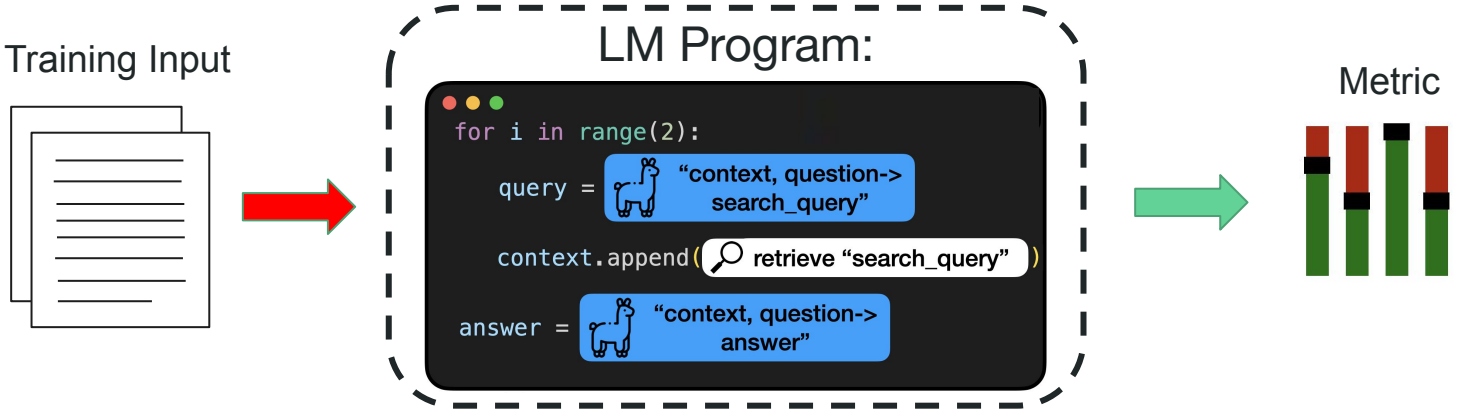
Step 1: Bootstrap Task Demonstrations



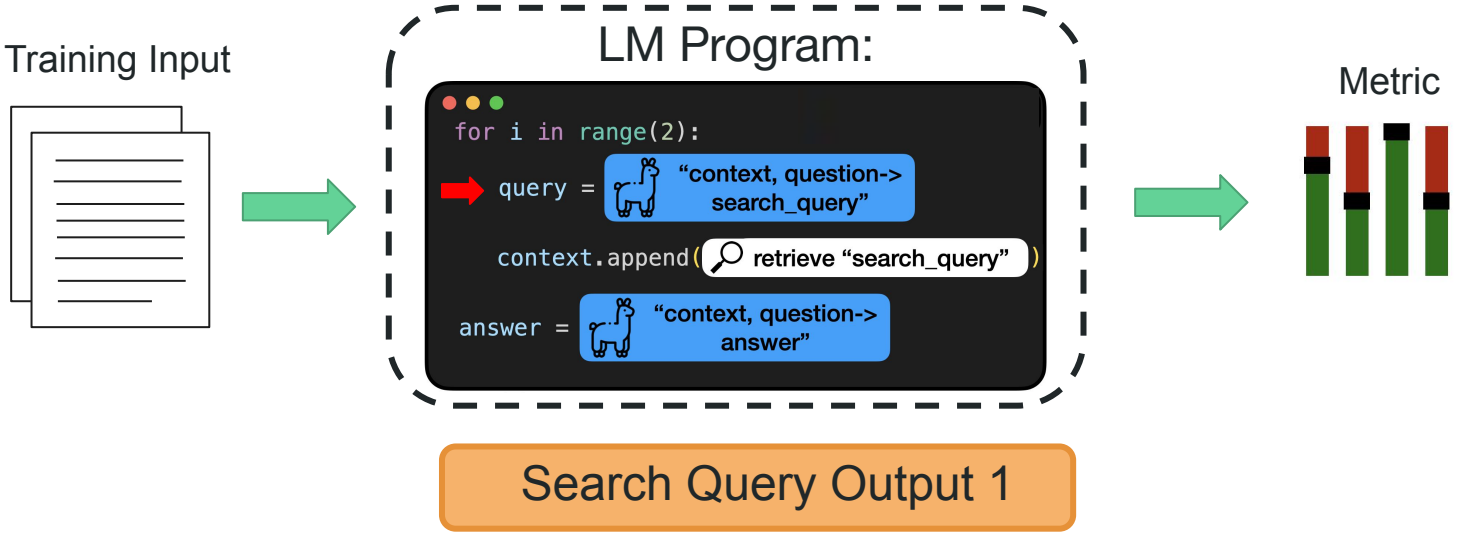
Step 1: Bootstrap Task Demonstrations



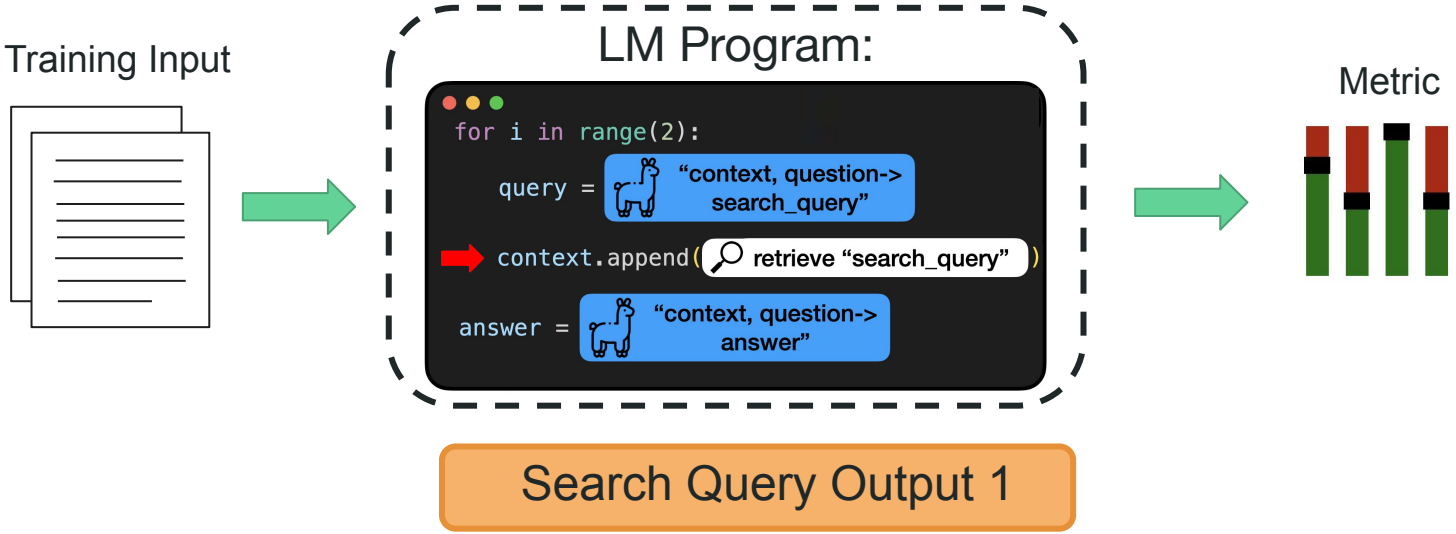
Step 1: Bootstrap Task Demonstrations



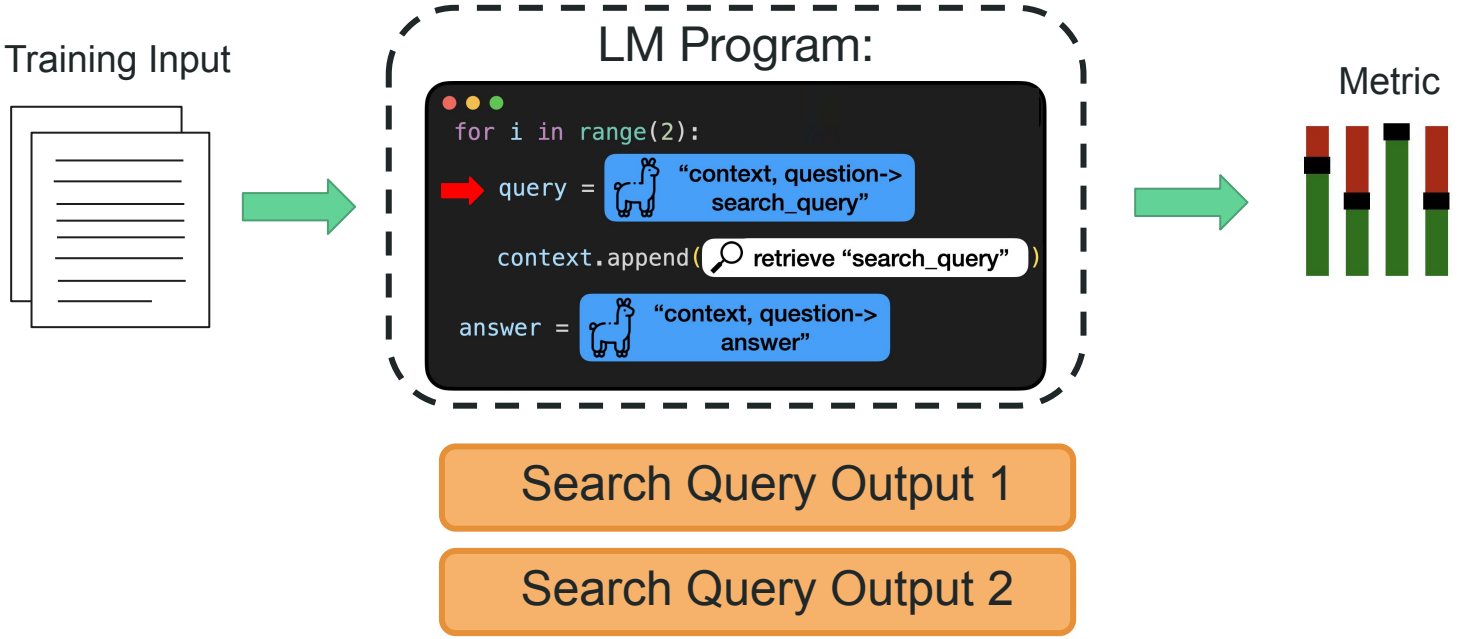
Step 1: Bootstrap Task Demonstrations



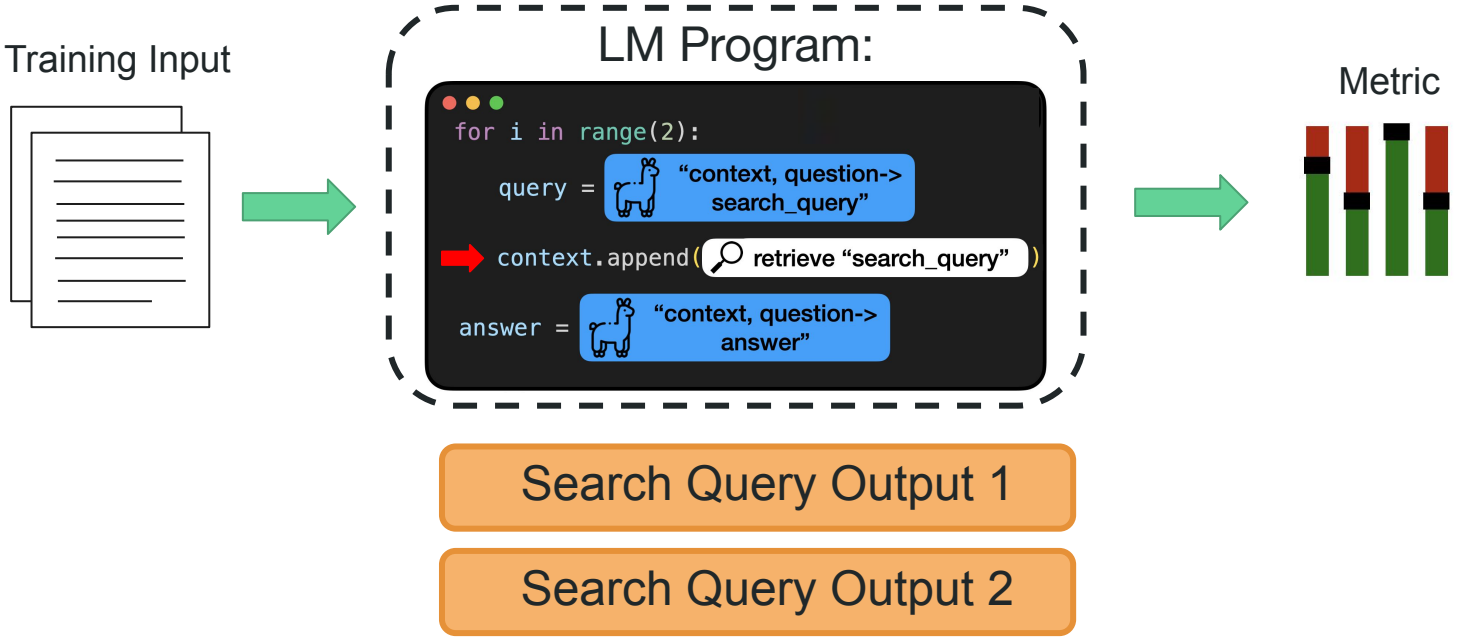
Step 1: Bootstrap Task Demonstrations



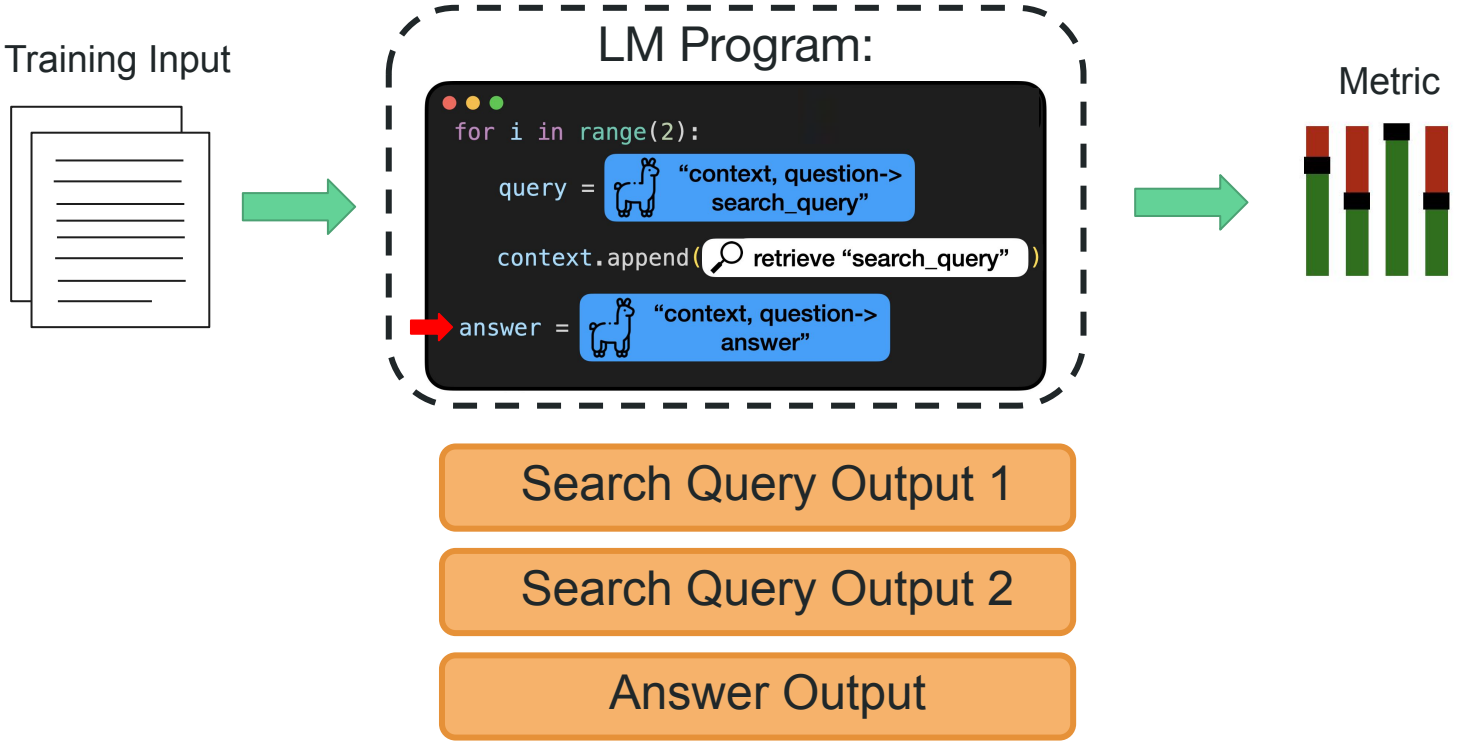
Step 1: Bootstrap Task Demonstrations



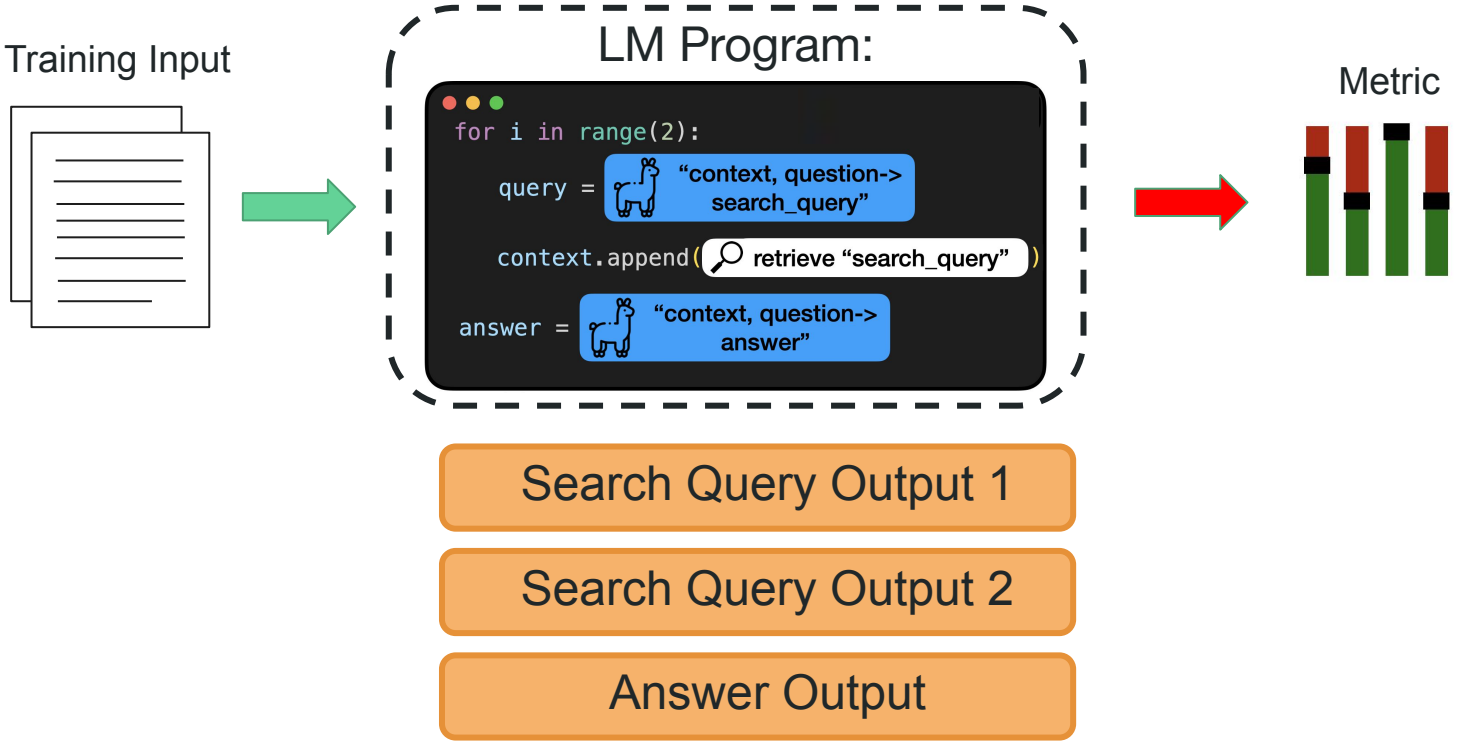
Step 1: Bootstrap Task Demonstrations



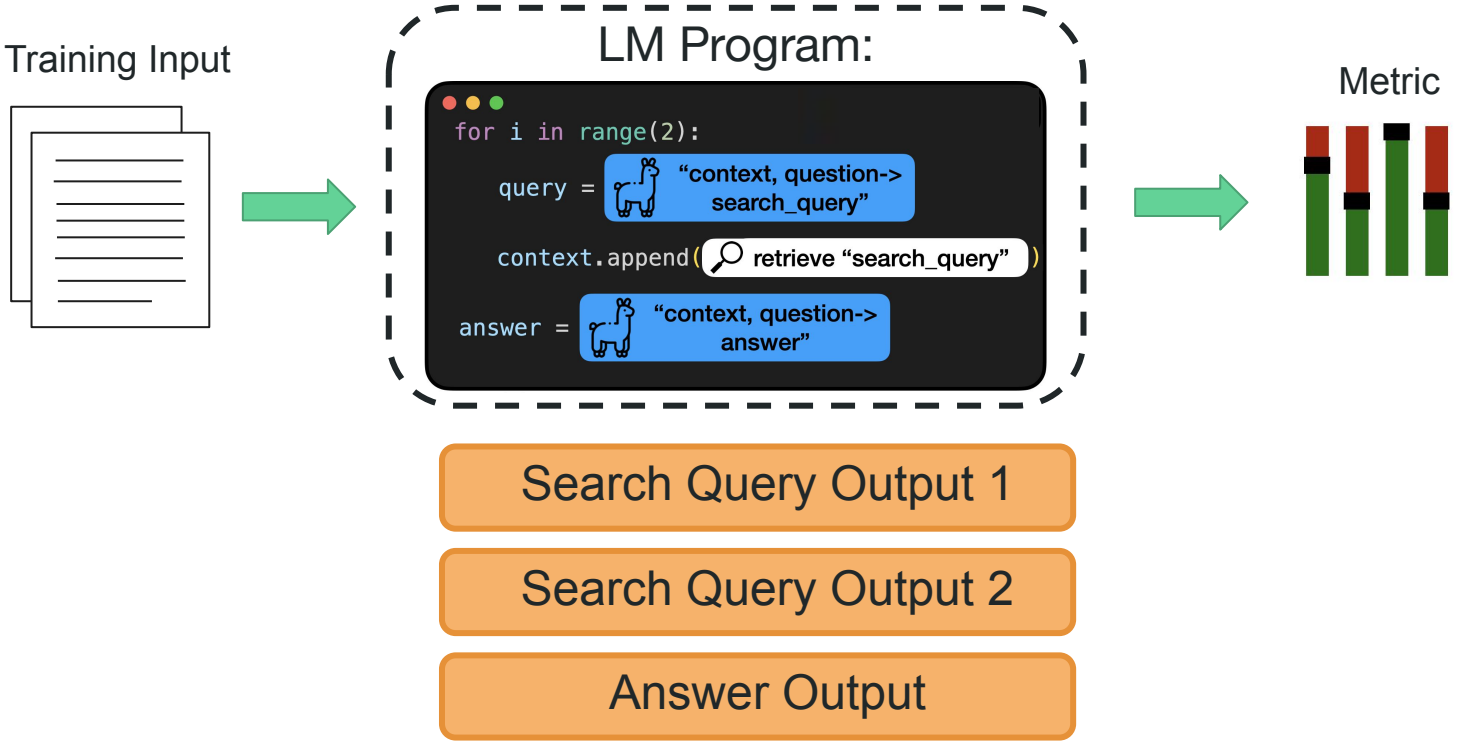
Step 1: Bootstrap Task Demonstrations



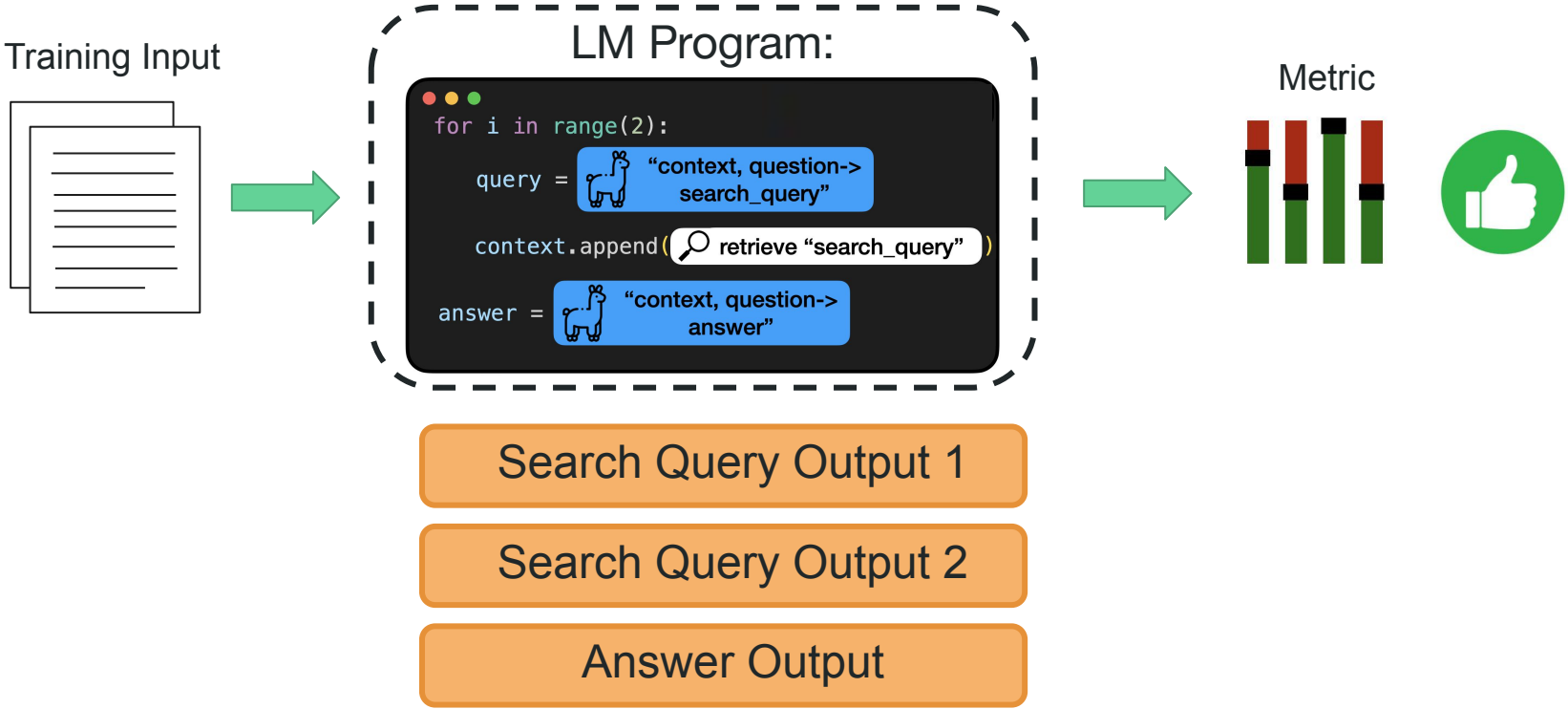
Step 1: Bootstrap Task Demonstrations



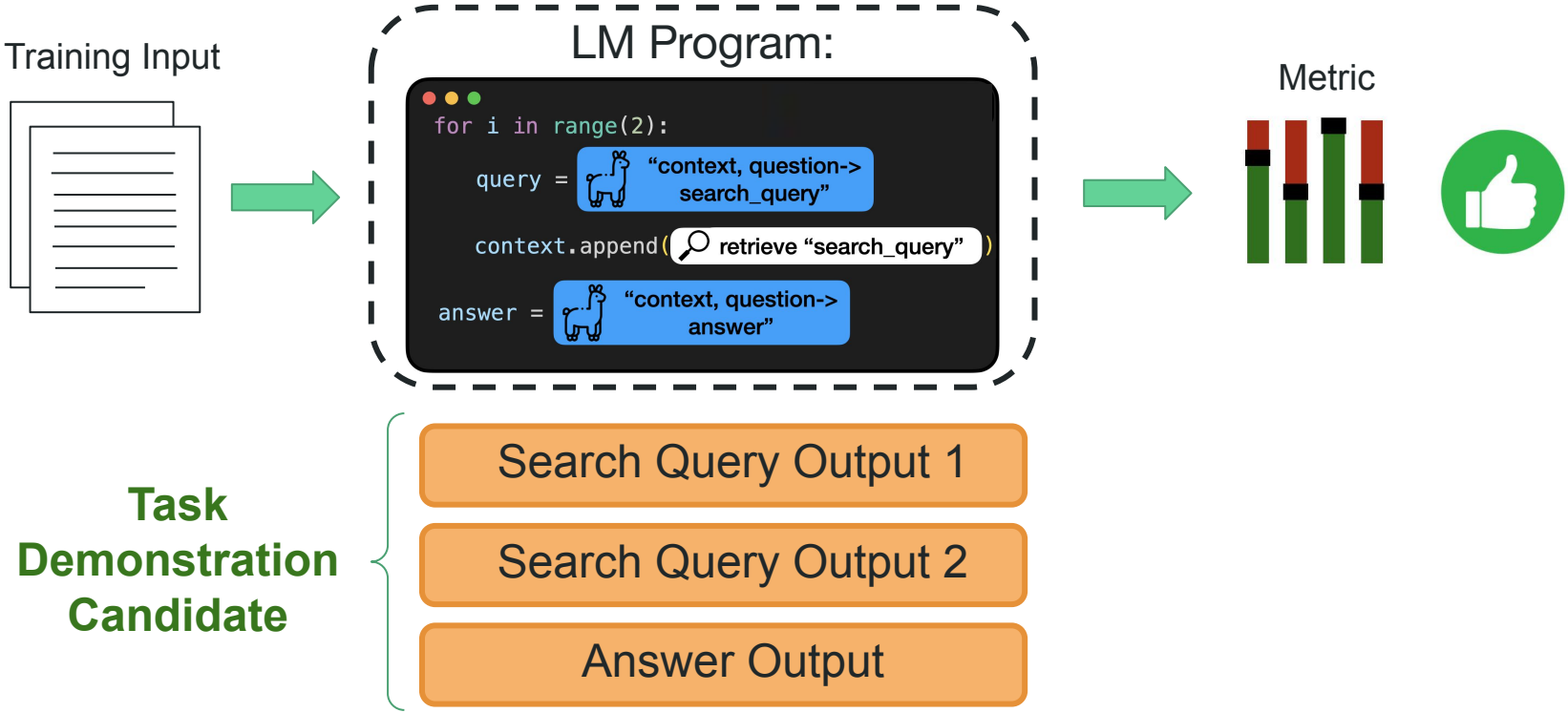
Step 1: Bootstrap Task Demonstrations



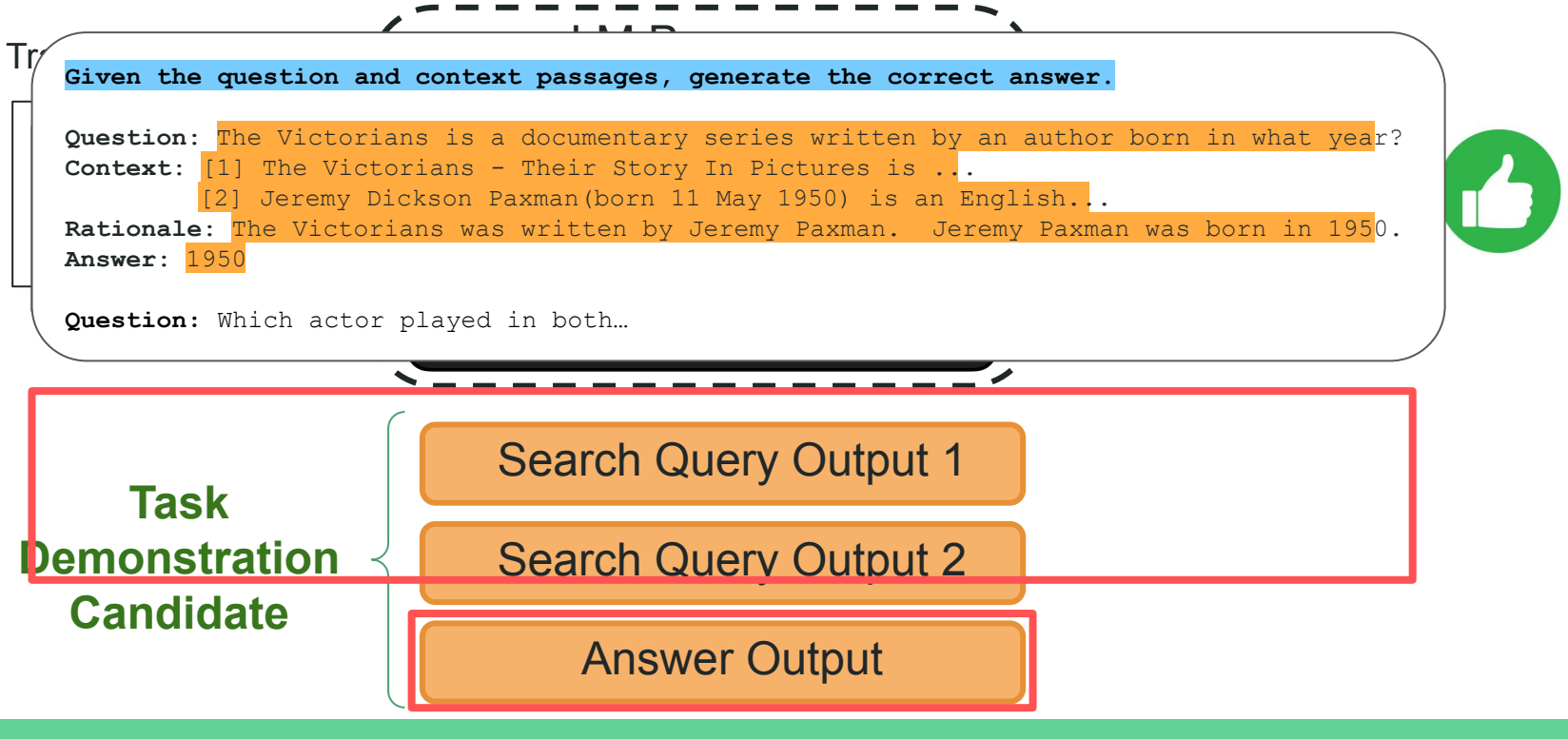
Step 1: Bootstrap Task Demonstrations



Step 1: Bootstrap Task Demonstrations



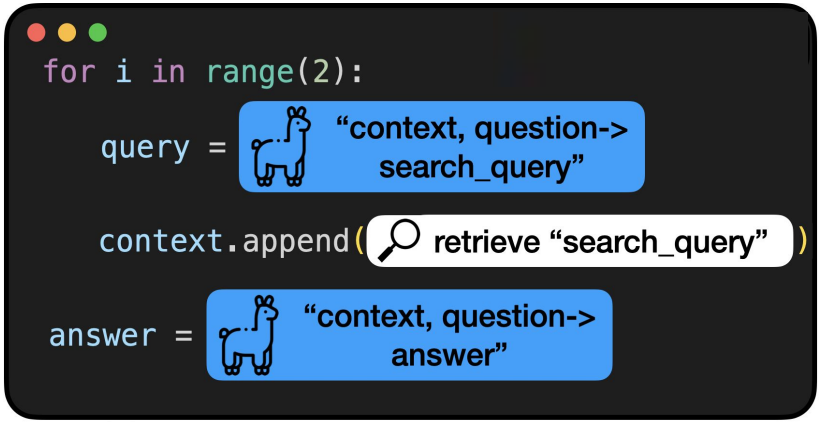
Step 1: Bootstrap Task Demonstrations



Step 2: Propose Instruction Candidates

Key Idea: Grounding (Understanding your Task)

Program Aware:



“This program is designed to solve tasks related to natural language processing, particularly in answering questions from a given context. It uses language models to generate search queries, answer questions, and rank passages based on their helpfulness in answering a given question.”

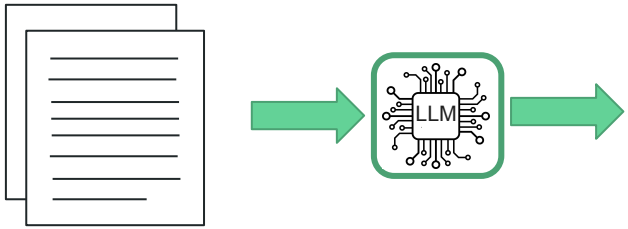
Step 2: Propose Instruction Candidates

Key Idea: Grounding (Understanding your Task)

Program Aware:

</>

Data Aware:



"The dataset contains trivia-style questions from a wide range of topics like music, film, history, and literature. Questions are well-structured and require specific information as answers, suggesting a focus on testing knowledge. The dataset's consistent format and emphasis on accuracy make it suitable for developing a trivia quiz application or knowledge testing platform."

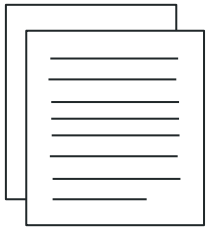
Step 2: Propose Instruction Candidates

Key Idea: Grounding (Understanding your Task)

Program Aware:

</>

Data Aware:



Demo Aware:

Search Query Output 1

Search Query Output 2

Answer Output

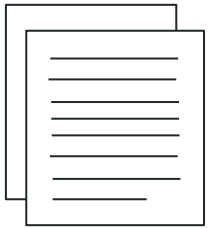
Step 2: Propose Instruction Candidates

Key Idea: Grounding (Understanding your Task)

Program Aware:

</>

Data Aware:



Demo Aware:

Prompting Tips:

"Don't be afraid to be creative."

"Include a high stakes scenario…"

"Provide the LLM with a persona"

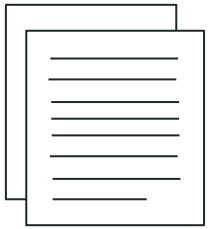
Step 2: Propose Instruction Candidates

Key Idea: Grounding (Understanding your Task)

Program Aware:

</>

Data Aware:



Demo Aware:

Prompting Tips:

Step 2: Propose Instruction Candidates

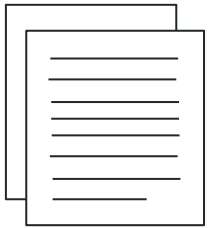
Key Idea: Grounding (Understanding your Task)

Program Aware:

Data Aware:

Demo Aware:

Prompting Tips:



Step 2: Propose Instruction Candidates

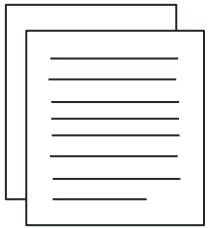
Key Idea: Grounding (Understanding your Task)

Program Aware:

Data Aware:

Demo Aware:

Prompting Tips:



Step 2: Propose Instruction Candidates

"Given a context containing information about a topic and a question related to that topic, generate a detailed and accurate answer to the question based on the provided context"

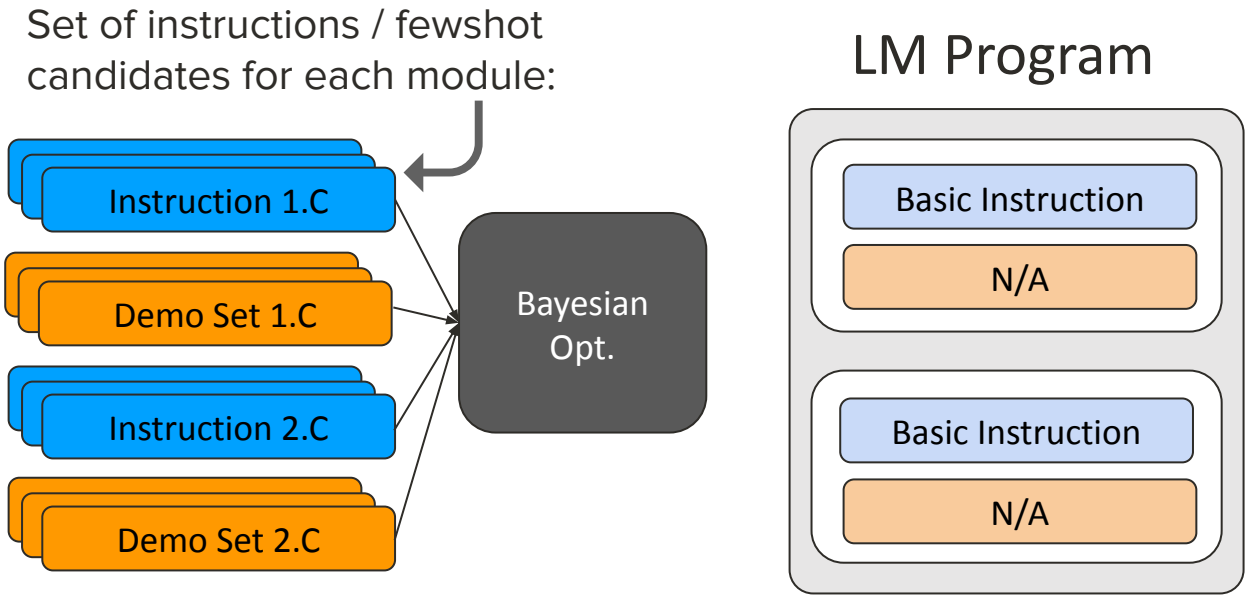
"Generate an answer to the given question based on the provided context and question."

"Use a language model to generate a detailed and accurate answer to a given question by providing the fields `context` and `question` and producing the field `answer`, ensuring that the answer is specific and relevant to the input context and question."

"Given the detailed `context` and the `question`, use the pipeline of language models to precisely generate the `answer`. The language model should carefully consider the specific details of the question and extract the most relevant information from the context in order to provide the accurate answer."

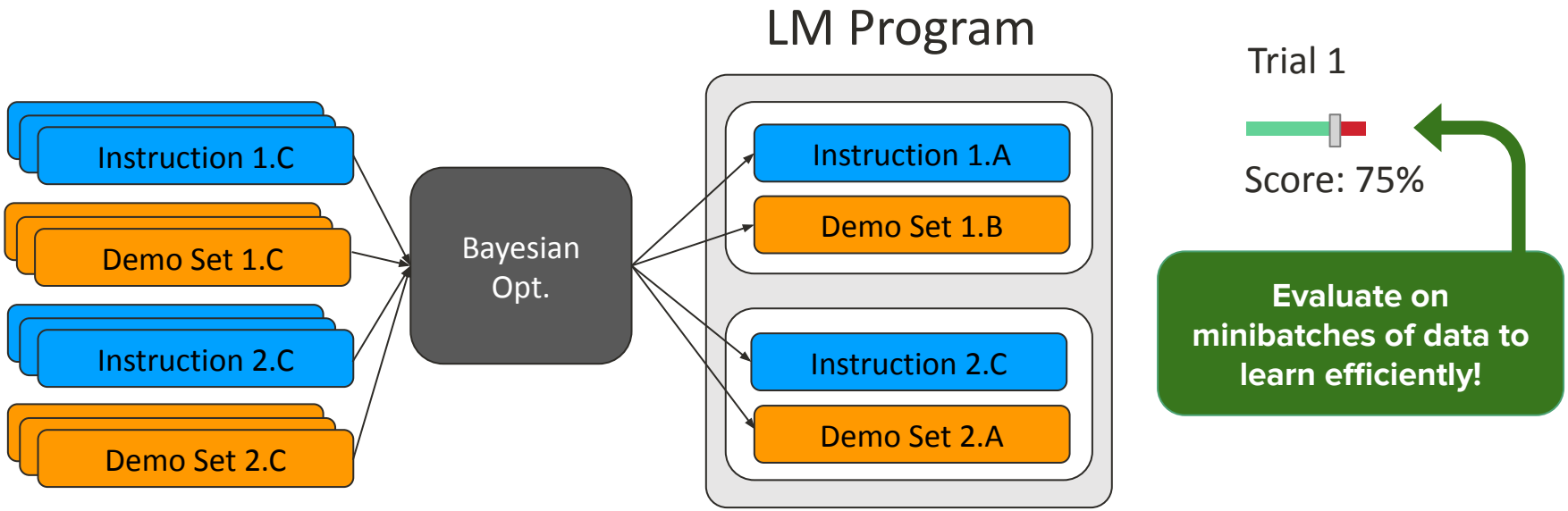
Step 3: Optimize the Combination of Both

Key Idea: MIPRO uses a Bayesian Surrogate Model for Credit Assignment



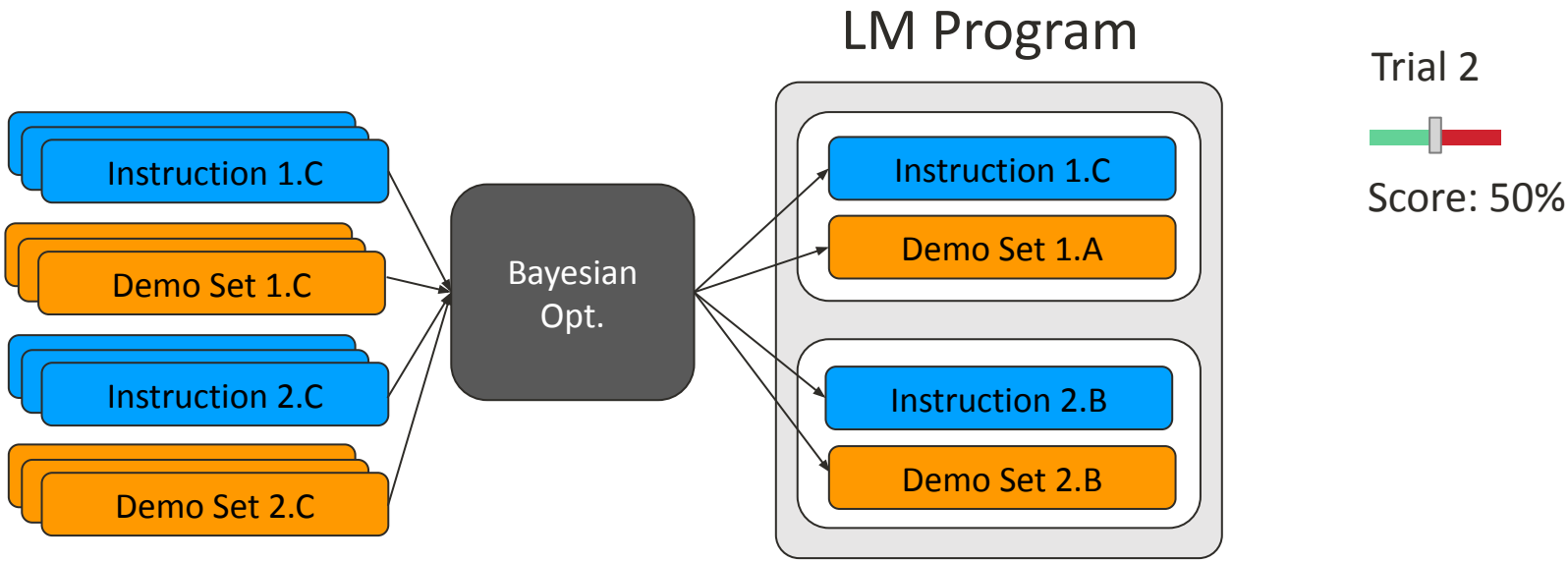
Step 3: Optimize the Combination of Both

Key Idea: MIPRO uses a Bayesian Surrogate Model for Credit Assignment



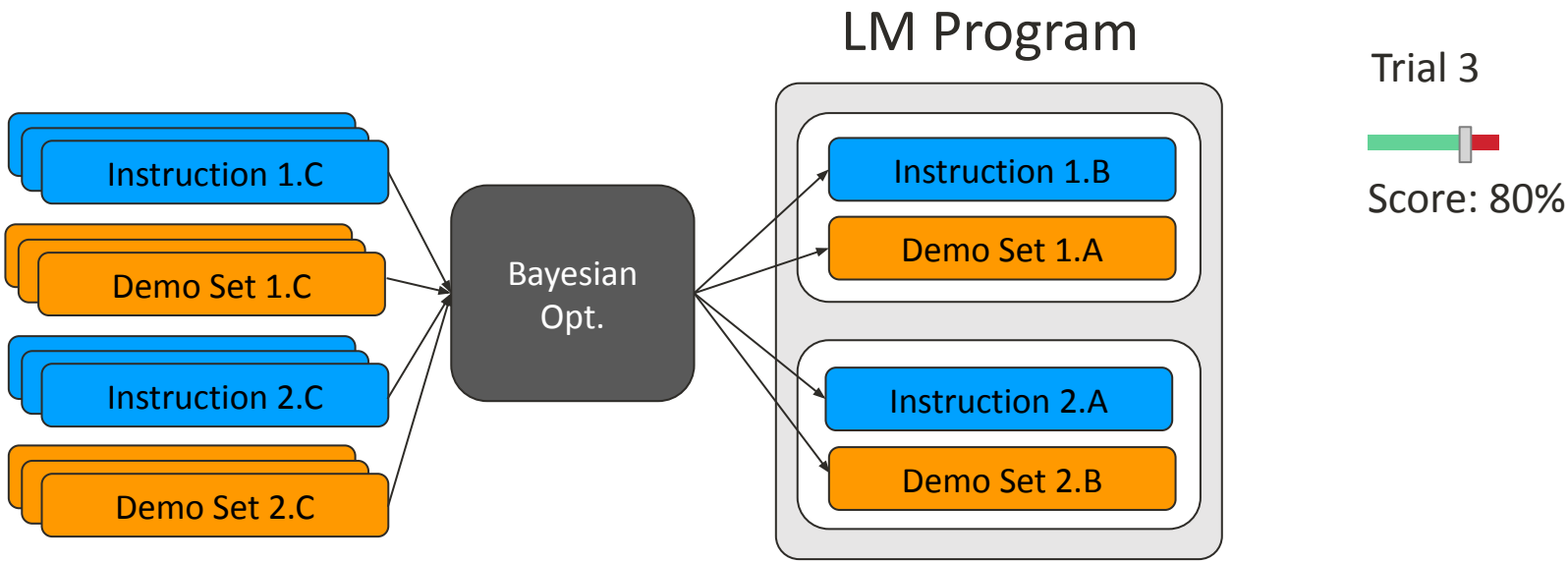
Step 3: Optimize the Combination of Both

Key Idea: MIPRO uses a Bayesian Surrogate Model for Credit Assignment



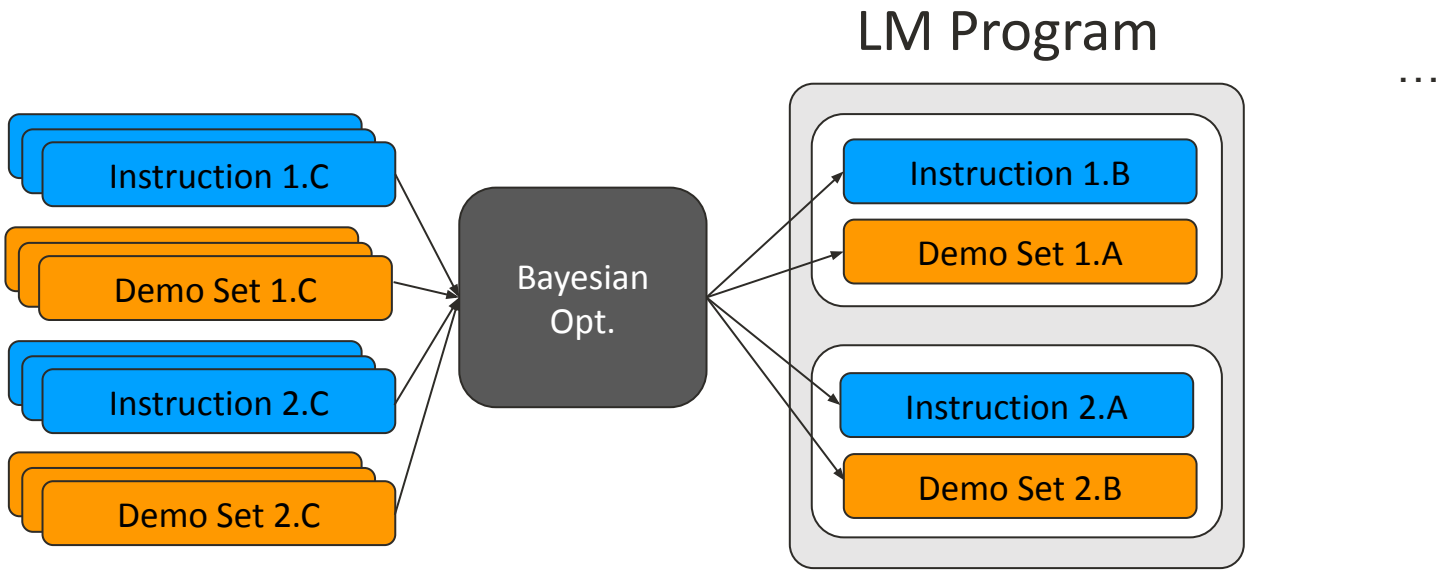
Step 3: Optimize the Combination of Both

Key Idea: MIPRO uses a Bayesian Surrogate Model for Credit Assignment



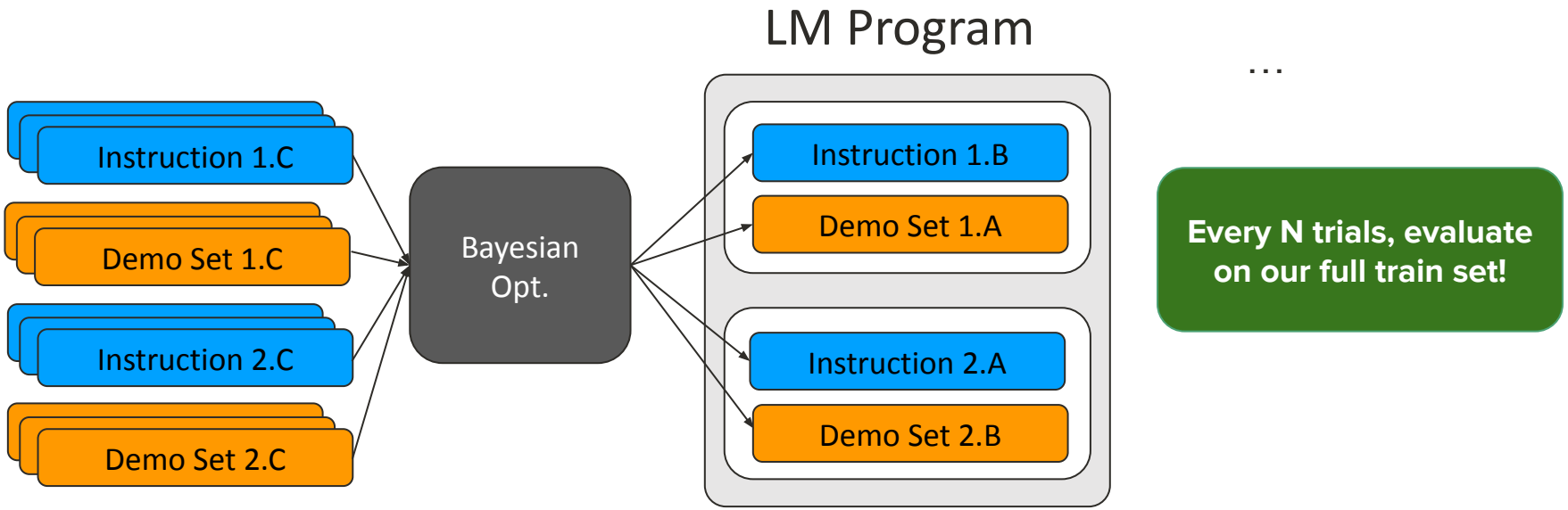
Step 3: Optimize the Combination of Both

Key Idea: MIPRO uses a Bayesian Surrogate Model for Credit Assignment



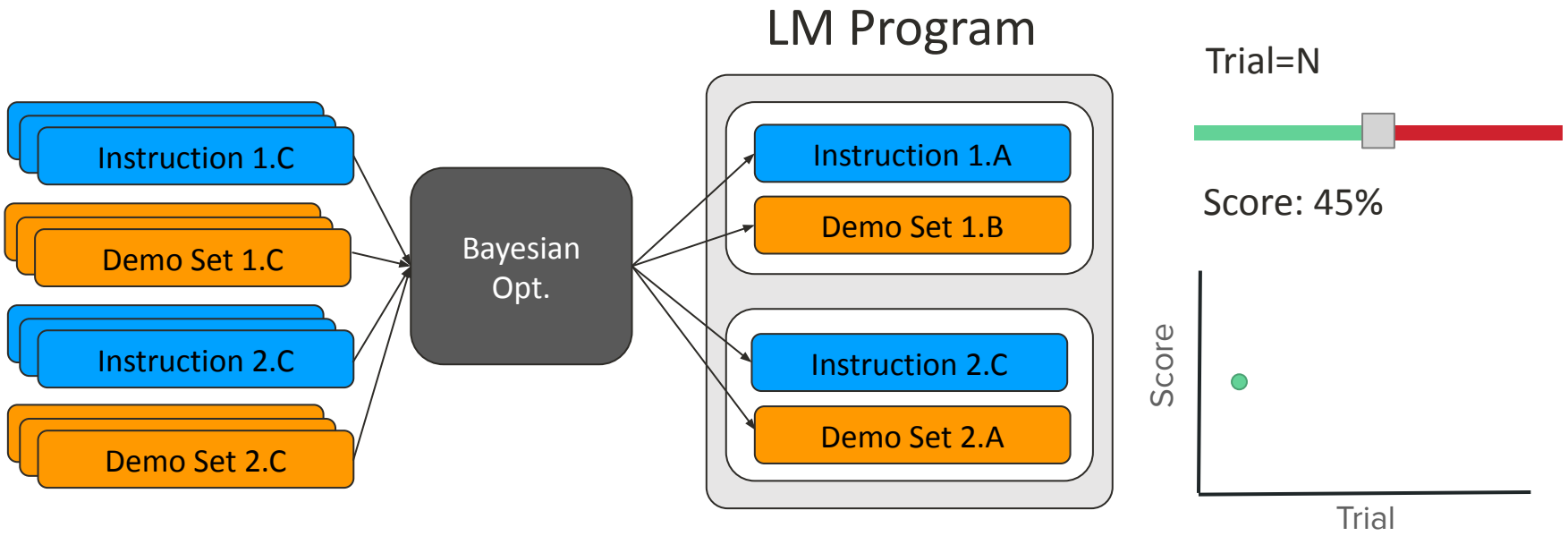
Step 3: Optimize the Combination of Both

Key Idea: MIPRO uses a Bayesian Surrogate Model for Credit Assignment



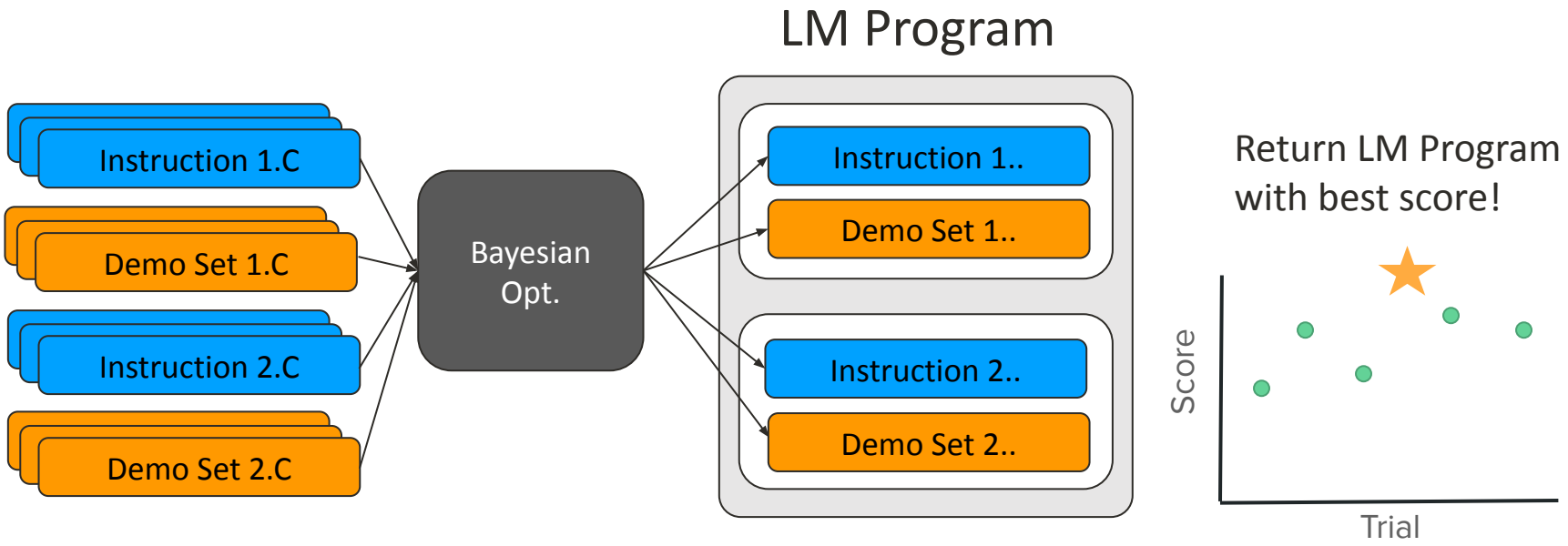
Step 3: Optimize the Combination of Both

Key Idea: MIPRO uses a Bayesian Surrogate Model for Credit Assignment



Step 3: Optimize the Combination of Both

Key Idea: MIPRO uses a Bayesian Surrogate Model for Credit Assignment



So how well does an optimizer like MIPROv2 work?

Enter LangProBe, the Language Model Program Benchmark

Benchmark	Task Type	Program	Modules	LM Calls	Metric
HotPotQA	Multi-Hop QA	Multi-Hop Retrieval	2	3	Exact Match
HotPotQA Conditional	Multi-Hop QA	Multi-Hop Retrieval	2	3	Custom
Iris	Classification	Chain of Thought	1	1	Accuracy
Iris-Typo	Classification	Chain of Thought	1	1	Accuracy
Heart Disease	Classification	Answer Ensemble	2	4	Accuracy
ScoNe	Natural Language Inference	Chain of Thought	1	1	Exact Match
HoVer	Multi-Hop Claim Verify	Multi-Hop Retrieval	4	4	Recall@21

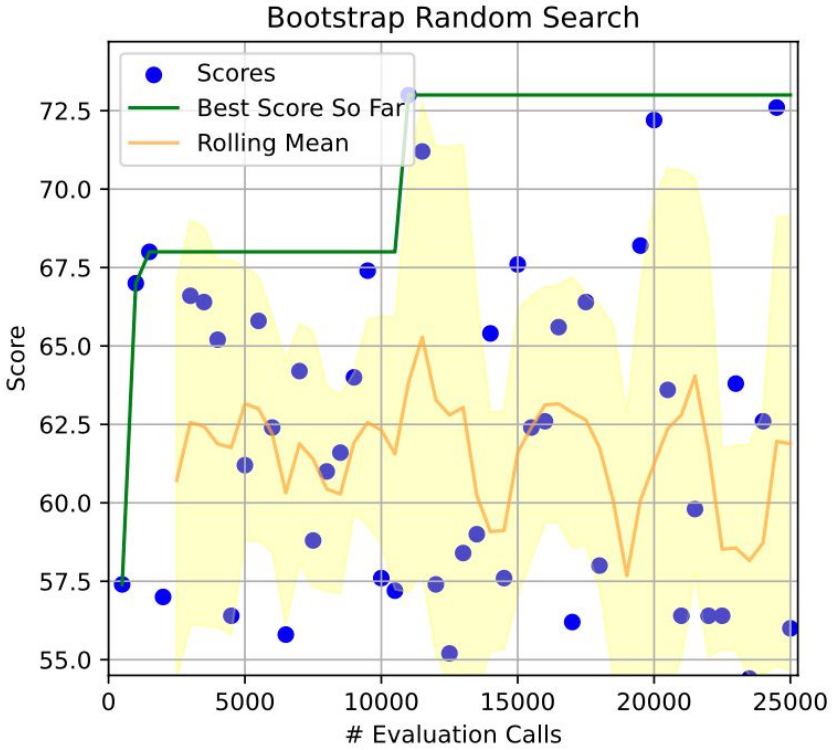
LangProBe is currently available on DSPy!

Optimizer	ScoNe	HotPotQA	HoVer	HotPotQA Cond.	Iris	Iris-Typo	Heart Disease
Instructions only (0-shot)							
N/A	57.0	35.4	30.2	13.8	46.4	34.7	23.3
Module-Level OPRO −G	70.0	36.0	30.0	–	–	–	–
Module-Level OPRO	69.1	41.9	37.1	–	–	–	–
0-Shot MIPRO	66.3	40.2	37.7	22.6	40.8	56.8	26.8
0-Shot MIPRO++	69.0	41.5	37.1	–	–	–	–

Optimizing instructions can deliver gains over baseline signatures. But there's no obvious "best" approach yet.

Optimizer	ScoNe	HotPotQA	HoVer	HotPotQA Cond.	Iris	Iris-Typo	Heart Disease
Instructions only (0-shot)							
N/A	57.0	35.4	30.2	13.8	46.4	34.7	23.3
Module-Level OPRO −G	70.0	36.0	30.0	–	–	–	–
Module-Level OPRO	69.1	41.9	37.1	–	–	–	–
0-Shot MIPRO	66.3	40.2	37.7	22.6	40.8	56.8	26.8
0-Shot MIPRO++	69.0	41.5	37.1	–	–	–	–
Demonstrations only (Few-shot)							
Bootstrap RS	74.9	48.6	42.0	16.4	95.2	58.9	78.4
Bayesian Bootstrap	75.4	49.2	44.6	–	–	–	–

Optimizing bootstrapped demonstrations is key!



The bootstrapped demonstrations we choose matters a lot!
Understanding why is an area for future research.

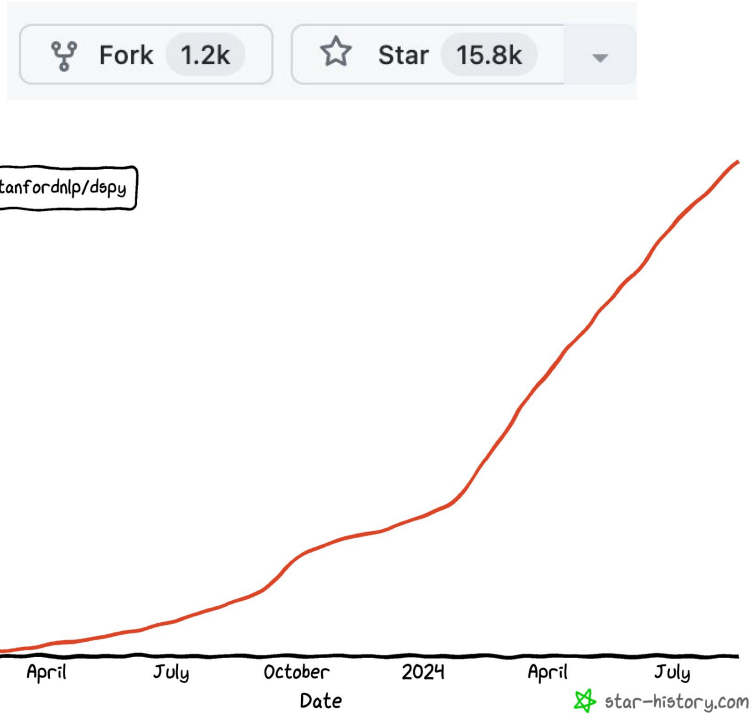
Optimizer	ScoNe	HotPotQA	HoVer	HotPotQA Cond.	Iris	Iris-Typo	Heart Disease
Instructions only (0-shot)							
N/A	57.0	35.4	30.2	13.8	46.4	34.7	23.3
Module-Level OPRO −G	70.0	36.0	30.0	–	–	–	–
Module-Level OPRO	69.1	41.9	37.1	–	–	–	–
0-Shot MIPRO	66.3	40.2	37.7	22.6	40.8	56.8	26.8
0-Shot MIPRO++	69.0	41.5	37.1	–	–	–	–
Demonstrations only (Few-shot)							
Bootstrap RS	74.9	48.6	42.0	16.4	95.2	58.9	78.4
Bayesian Bootstrap	75.4	49.2	44.6	–	–	–	–
Both (Few-shot)							
MIPRO	74.6	49.0	44.7	28.4	98.4	69.1	75.2

Optimizing both instructions and demonstrations via MIPROv2 is a often the most effective approach!

Optimizer	ScoNe	HotPotQA	HoVer	HotPotQA Cond.	Iris	Iris-Typo	Heart Disease
Instructions only (0-shot)							
N/A	57.0	35.4	30.2	13.8	46.4	34.7	23.3
Module-Level OPRO −G	70.0	36.0	30.0	–	–	–	–
Module-Level OPRO	69.1	41.9	37.1	–	–	–	–
0-Shot MIPRO	66.3	40.2	37.7	22.6	40.8	56.8	26.8
0-Shot MIPRO++	69.0	41.5	37.1	–	–	–	–
Demonstrations only (Few-shot)							
Bootstrap RS	74.9	48.6	42.0	16.4	95.2	58.9	78.4
Bayesian Bootstrap	75.4	49.2	44.6	–	–	–	–
Both (Few-shot)							
MIPRO	74.6	49.0	44.7	28.4	98.4	69.1	75.2

The impact of optimizing instructions (rather than demonstrations) is more visible in tasks that have many isolated conditional rules.

Lots of people are using DSPy & MIPRO!



… from both academia & industry!

Stanford University, MIT, Berkeley University of California, University of Toronto, Georgia Tech, University of Maryland, Google Labs, Walmart, databricks, vmware, Amazon, snowflake®

(& more…)

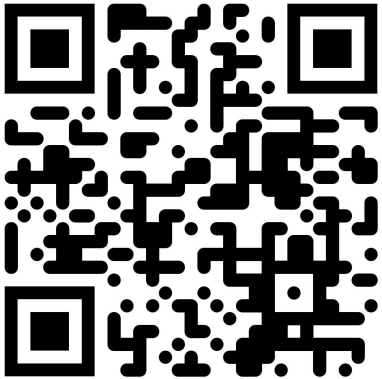
Learn more!

📄 **Read our paper!**

Arxiv

⚡**Getting started with MIPROv2**

Thread + notebook



🧩 **Getting started w/ DSPy**

DSPy tutorial

Thank you!