Towards Efficient Data Wrangling with LLMs using Code Generation

Effy Xue Li, Till Döhmen
University of Amsterdam, MotherDuck
DBDBD’24

Date 11/2024

Data Wrangling is Tedious

A data scientist is working on a table with 1 million rows, and she is facing following tasks…

Prompt used: A duck plumber who looks bored

Data Wrangling is Tedious

A data scientist is working on a table with 1 million rows, and she is facing following tasks…

Prompt used: A duck plumber who looks bored

	Phone	City
1	212/582-7200	

	Phone	City
1	212/582-7200	NYC

Data Imputation

Data Wrangling is Tedious

A data scientist is working on a table with 1 million rows, and she is facing following tasks…

Prompt used: A duck plumber who looks bored

	Phone	City
1	212/582-7200	

	Hospital	Error?
1	medixal xenter	

	Phone	City
1	212/582-7200	NYC

	Hospital	Error?
1	medixal xenter	Yes

Data Imputation Error Detection

Data Wrangling is Tedious

A data scientist is working on a table with 1 million rows, and she is facing following tasks…

Prompt used: A duck plumber who looks bored

	Phone	City
1	212/582-7200	

	Hospital	Error?
1	medixal xenter	

	Celsius	Fahrenheit
1	7	

	Phone	City
1	212/582-7200	NYC

	Hospital	Error?
1	medixal xenter	Yes

	Celsius	Fahrenheit
1	7	44.6

Data Imputation

Error Detection

Data Transformation

Data Wrangling is Tedious

A data scientist is working on a table with 1 million rows, and she is facing following tasks…

Prompt used: A duck plumber who looks bored

	Phone	City
1	212/582-7200	

	Phone	City
1	212/582-7200	NYC

Data Imputation

	Hospital	Error?
1	medixal xenter	

	Hospital	Error?
1	medixal xenter	Yes

Error Detection

	Celsius	Fahrenheit
1	7	

	Celsius	Fahrenheit
1	7	44.6

Data Transformation

	Beer_Name	BV
1	IJwit IPA	6.50 %

	Beer_Name	BV
1	IJwit IPA	0.50 %

same?

No

Entity Matching

Data Wrangling is Tedious

A data scientist is working on a table with 1 million rows, and she is facing following tasks…

Prompt used: A duck plumber who looks bored

How can we make this easier?

	Phone	City
1	212/582-7200	

	Phone	City
1	212/582-7200	NYC

Data Imputation

	Hospital	Error?
1	medixal xenter	Yes

Error Detection

	Celsius	Fahrenheit
1	7	44.6

Data Transformation

	Beer_Name	BV
1	IJwit IPA	6.50 %

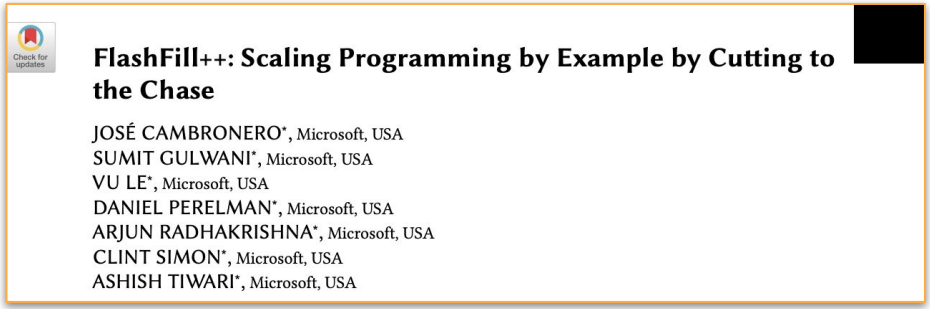
	Beer_Name	BV
1	IJwit IPA	0.50 %

same?

No

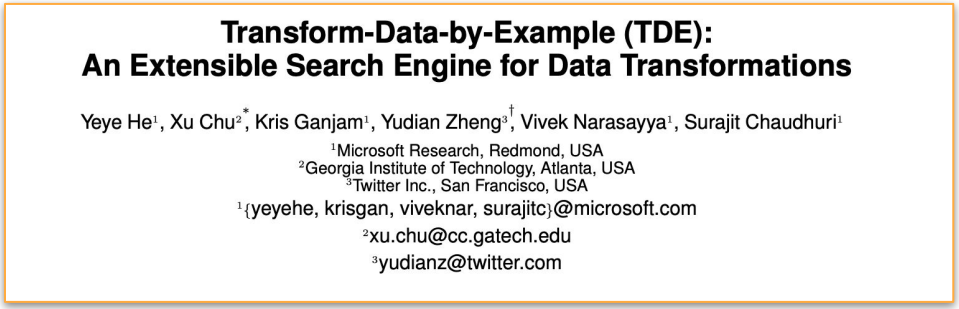
Entity Matching

Programing by Example (PBE)



FlashFill++: Scaling Programming by Example by Cutting to the Chase

JOSÉ CAMBRONERO*, Microsoft, USA
SUMIT GULWANI*, Microsoft, USA
VU LE*, Microsoft, USA
DANIEL PERELMAN*, Microsoft, USA
ARJUN RADHAKRISHNA*, Microsoft, USA
CLINT SIMON*, Microsoft, USA
ASHISH TIWARI*, Microsoft, USA



Transform-Data-by-Example (TDE): An Extensible Search Engine for Data Transformations

Yeye He[1], Xu Chu[2]*, Kris Ganjam[1], Yudian Zheng[3]†, Vivek Narasayya[1], Surajit Chaudhuri[1]
[1]Microsoft Research, Redmond, USA
[2]Georgia Institute of Technology, Atlanta, USA
[3]Twitter Inc., San Francisco, USA
[1]{yeyehe, krisgan, viveknar, surajitc}@microsoft.com
[2]xu.chu@cc.gatech.edu
[3]yudianz@twitter.com

- Domain-specific Language search space
- Search
- Rank
- Solution

Programming by Example is not flexible

- Have to define a program search space (often DSL).
-
-
-
-

Programming by Example is not flexible

- Have to define a program search space (often DSL).
- Solves only **a limited number** of tasks.
-
-
-

Programming by Example is not flexible

- Have to define a program search space (often DSL).
- Solves only **a limited number** of tasks.
- Struggling at semantically challenging tasks.
-
-

Programming by Example is not flexible

- Have to define a program search space (often DSL).
- Solves only **a limited number** of tasks.
- Struggling at semantically challenging tasks.
- Do not take **natural language instructions** as inputs.
-

Programming by Example is not flexible

- Have to define a program search space (often DSL).
- Solves only **a limited number** of tasks.
- Struggling at semantically challenging tasks.
- Do not take **natural language instructions** as inputs.
- …

LLMs are Great for Data Wrangling

Input table

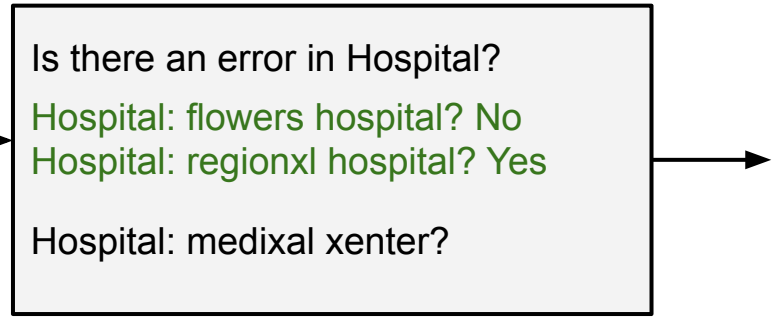
	Hospital
1	medixal xenter

Can Foundation Models Help Wrangle Data? Narayan et al. VLDB 23.

LLMs are Great for Data Wrangling

Input table

	Hospital
1	**medixal xenter**

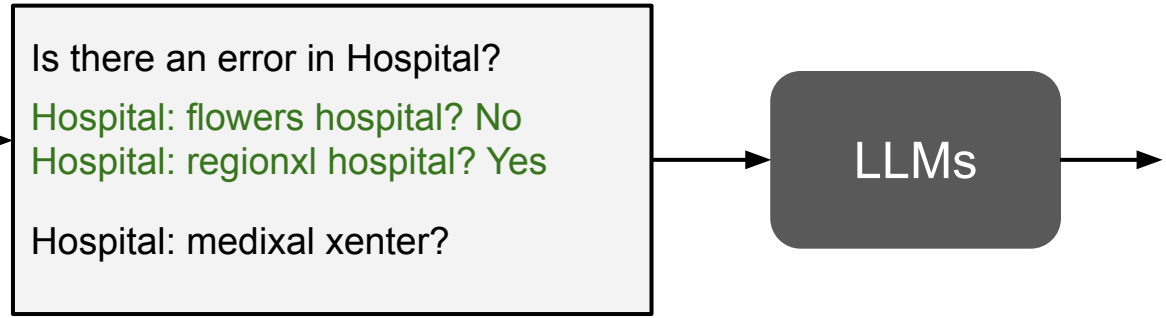


Can Foundation Models Help Wrangle Data? Narayan et al. VLDB 23.

LLMs are Great for Data Wrangling

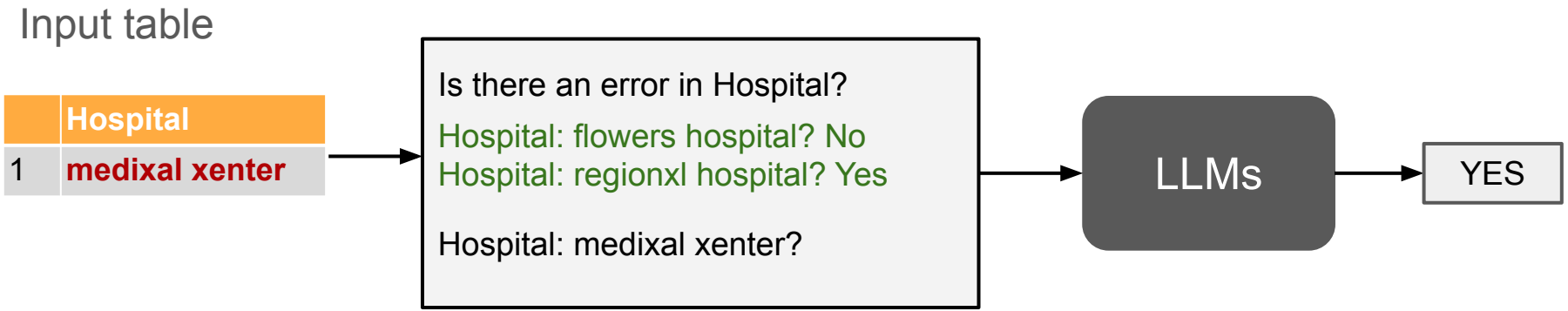
Input table

	Hospital
1	medixal xenter



Can Foundation Models Help Wrangle Data? Narayan et al. VLDB 23.

LLMs are Great for Data Wrangling

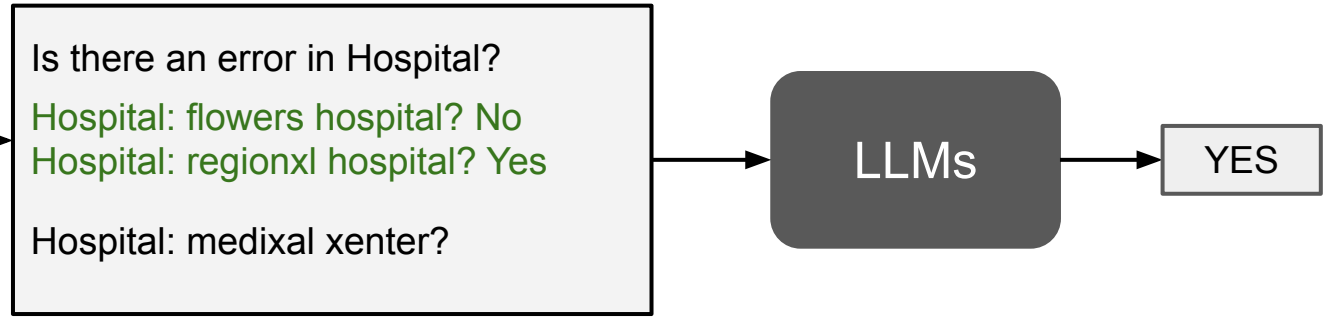


Can Foundation Models Help Wrangle Data? Narayan et al. VLDB 23.

LLMs are Great for Data Wrangling

Input table

	Hospital
1	medixal xenter



It performs LLM on a **per-row** basis (LLMPR).

Can Foundation Models Help Wrangle Data? Narayan et al. VLDB 23.

LLMs are ~~Great~~ Costly for Data Wrangling

LLMs are ~~Great~~ Costly for Data Wrangling

If we have a 1-million-rows-table, processing through it once will cost *…

	Time(with 100 concurrent calls)	Price
GPT-3.5	1.01 Hours	$ 12.5
GPT-4	**2.72 Hours**	**$ 600**
GPT-4o	-	$ 125

* Assuming input average token size is 10, and output is 5.

LLMs are ~~Great~~ Costly for Data Wrangling

If we have a 1-million-rows-table, processing through it once will cost *…

	Time(with 100 concurrent calls)	Price
GPT-3.5	1.01 Hours	$ 12.5
GPT-4	**2.72 Hours**	**$ 600**
GPT-4o	-	$ 125

And many other issues …

- **Transparency**
- **Reproducibility**
- **Privacy**
- …

* Assuming input average token size is 10, and output is 5.

Prompt: a duck plumber looking for smart tools

How can we make automated data wrangling faster

Prompt: a duck plumber looking for smart tools

How can we make automated data wrangling faster, cheaper

Prompt: a duck plumber looking for smart tools

How can we make automated data wrangling faster, cheaper, stronger

Prompt: a duck plumber looking for smart tools

How can we make automated data wrangling faster, cheaper, stronger(more reliable and more generic)?

Prompt: a duck plumber looking for smart tools

How can we make automated data wrangling faster, cheaper, stronger(more reliable and more generic)?

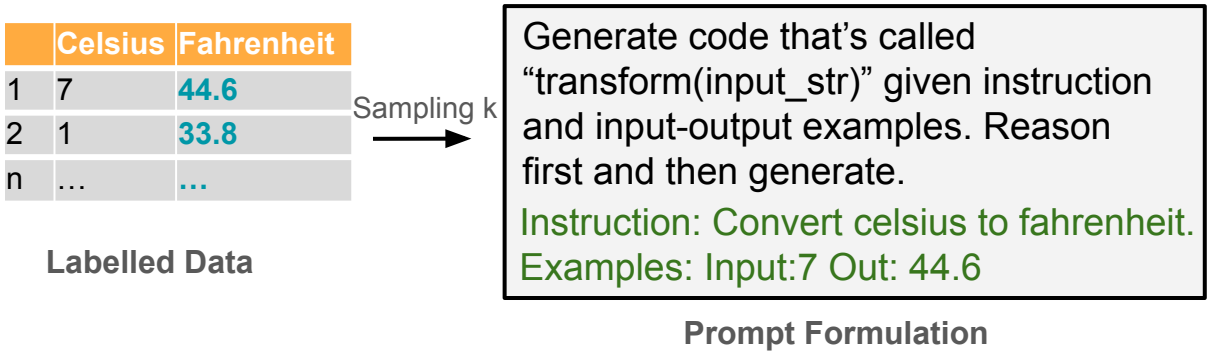
-> Can we combine PBE and LLMPR, using LLM to generate code for data wrangling?

Code Generation Framework

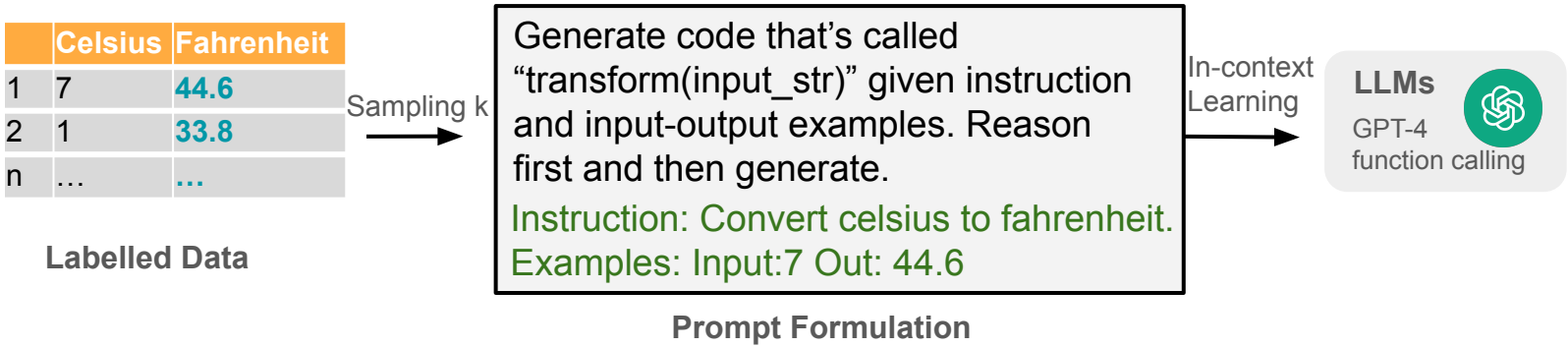
	Celsius	Fahrenheit
1	7	**44.6**
2	1	**33.8**
n	…	**…**

Labelled Data

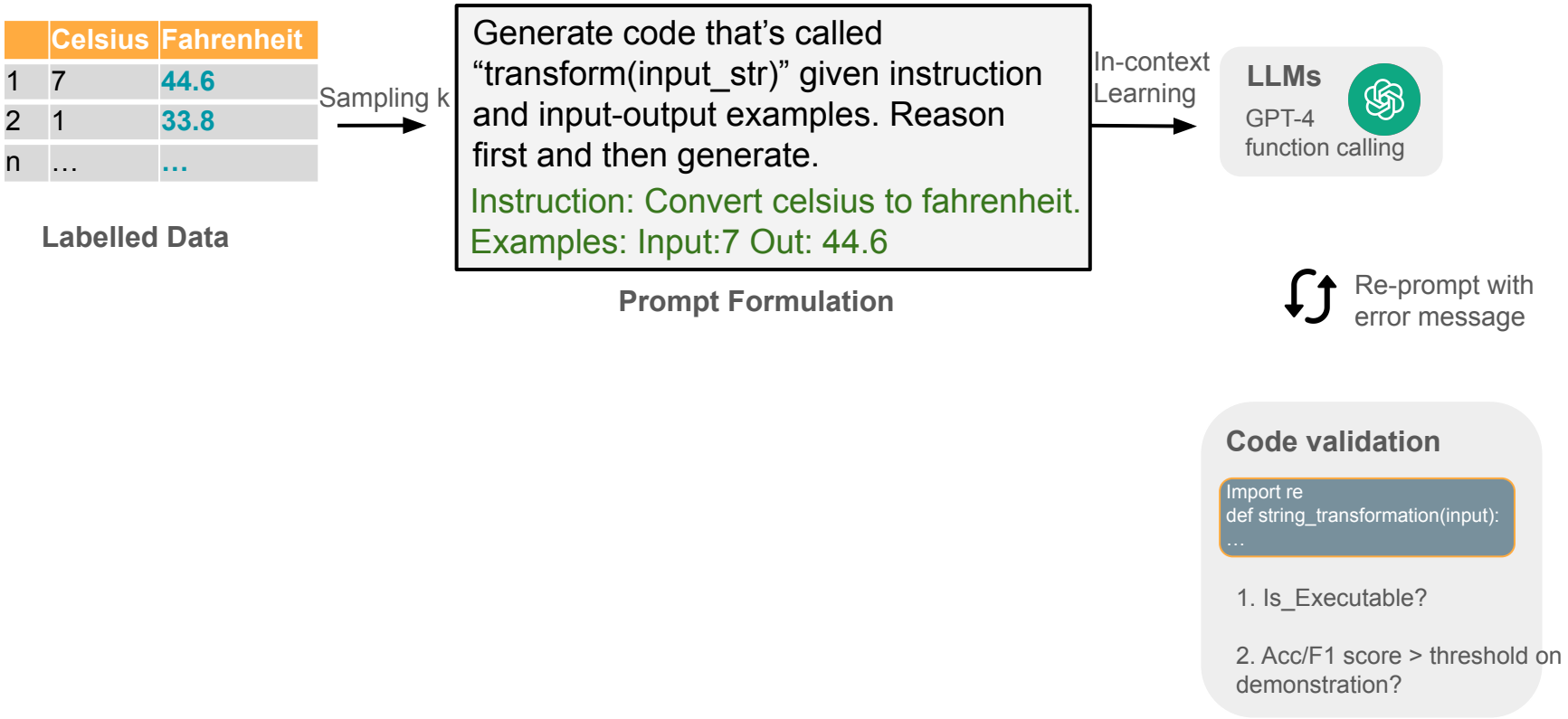
Code Generation Framework



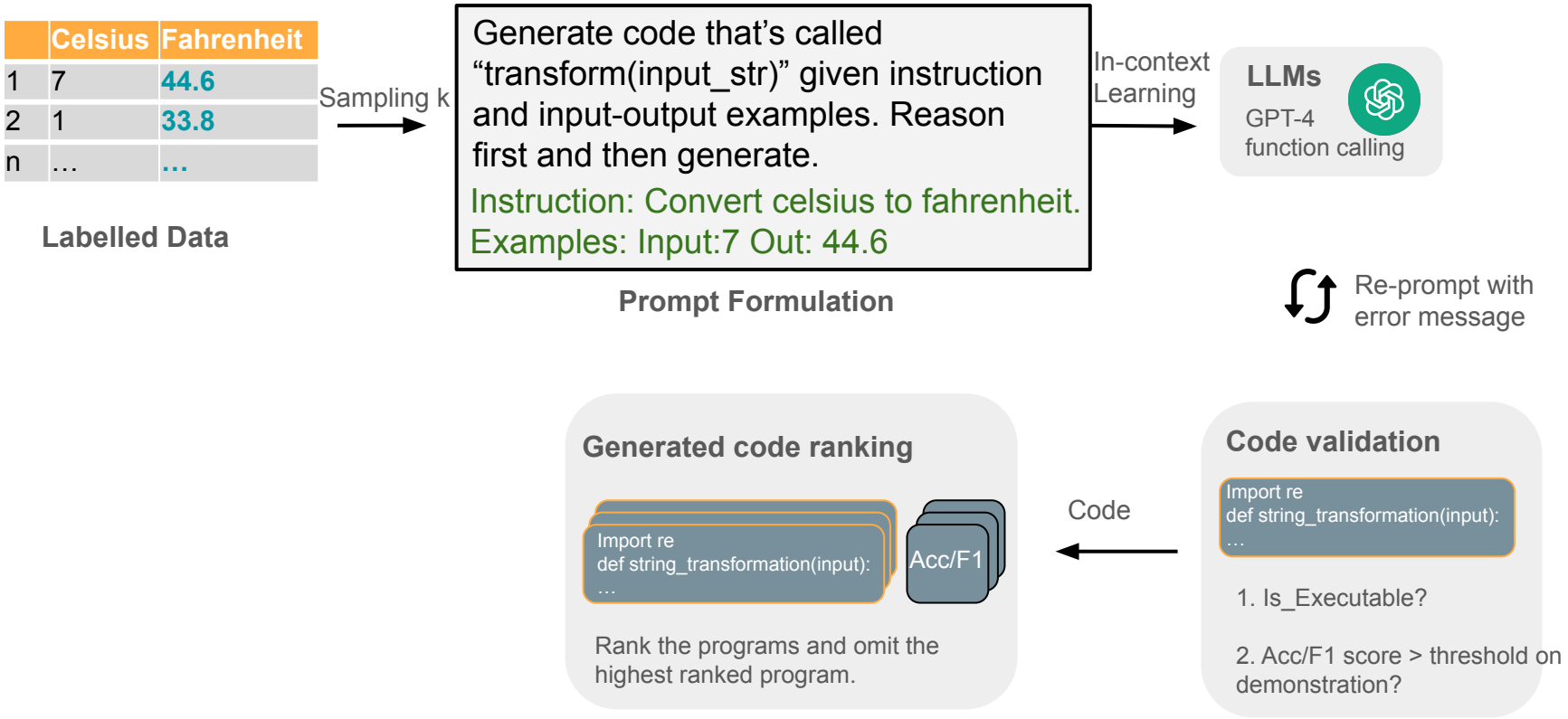
Code Generation Framework



Code Generation Framework



Code Generation Framework



Code Generation Framework

	Celsius	Fahrenheit
1	7	44.6
2	1	33.8
n	…	…

Labelled Data

Sampling k

Generate code that's called "transform(input_str)" given instruction and input-output examples. Reason first and then generate.

Instruction: Convert celsius to fahrenheit.
Examples: Input:7 Out: 44.6

Prompt Formulation

In-context Learning

LLMs
GPT-4
function calling

Re-prompt with error message

```
import re
def string_transformation(input_string):
    # Extract the numeric value from the input string
    celsius = float(re.search(r'\d+', input_string).group())
    # Convert Celsius to Fahrenheit
    fahrenheit = (celsius * 9/5) + 32
    # Format the result into the desired output string
    return f'{fahrenheit} Fahrenheit'
```

Generated code

N trials

Generated code ranking

Import re
def string_transformation(input):
…

Acc/F1

Rank the programs and omit the highest ranked program.

Code

Code validation

Import re
def string_transformation(input):
…

1. Is_Executable?

2. Acc/F1 score > threshold on demonstration?

We evaluate on benchmarks of these tasks:

	Phone	City
1	212/582-7200	

	Phone	City
1	212/582-7200	NYC

Data Imputation

	Hospital	Error?
1	medixal xenter	

	Hospital	Error?
1	medixal xenter	Yes

Error Detection

	Celsius	Fahrenheit
1	7	

	Celsius	Fahrenheit
1	7	44.6

Data Transformation

	Beer_Name	BV
1	IJwit IPA	6.50 %

	Beer_Name	BV
1	IJwit IPA	0.50 %

same?

No

Entity Matching

Preliminary Results on Python Generation

Entity Matching, Data Imputation, Error Detection

Task	Dataset	LLMPR[9]	Code Generation (Ours)
EM	Fodors-Zagats	100	95.5
EM	Beer	100	75.0
EM	DBLP-ACM	96.6	19.7
EM	DBLP-GoogleScholar	83.8	69.7
EM	Amazon-Google	63.5	42.1
EM	iTunes-Amazon	98.2	70.0
EM	Walmart-Amazon	87.0	25.5
DI	Buy	98.5	84.6
DI	Restaurant	88.4	50
ED	Hospital	97.8	23.5
ED	Adult	99.1	100*

Data Transformation

Dataset	PBE [4]	LLMPR [9]	Code Generation (Ours)
BingQL-semantics	32.0	54.0	**91.6**
BingQL-Unit	**96.0**	N/A	95.0
Stack-overflow	63.0	65.3	**87.4**
FF-GR-Trifacta	**91.0**	N/A	83.7
Head cases	**82.0**	N/A	74.6
Average	72.8	N/A	**86.46**

Preliminary Results on Python Generation

Entity Matching, Data Imputation, Error Detection

Task	Dataset	LLMPR[9]	Code Generation (Ours)
EM	Fodors-Zagats	100	95.5
EM	Beer	100	75.0
EM	DBLP-ACM	96.6	19.7
EM	DBLP-GoogleScholar	83.8	69.7
EM	Amazon-Google	63.5	42.1
EM	iTunes-Amazon	98.2	70.0
EM	Walmart-Amazon	87.0	25.5
DI	Buy	98.5	84.6
DI	Restaurant	88.4	50
ED	Hospital	97.8	23.5
ED	Adult	99.1	100*

Data Transformation

Dataset	PBE [4]	LLMPR [9]	Code Generation (Ours)
BingQL-semantics	32.0	54.0	**91.6**
BingQL-Unit	**96.0**	N/A	95.0
Stack-overflow	63.0	65.3	**87.4**
FF-GR-Trifacta	**91.0**	N/A	83.7
Head cases	**82.0**	N/A	74.6
Average	72.8	N/A	**86.46**

- **Evaluate based on rows.**

Preliminary Results on Python Generation

Entity Matching, Data Imputation, Error Detection

Task	Dataset	LLMPR[9]	Code Generation (Ours)
EM	Fodors-Zagats	100	95.5
EM	Beer	100	75.0
EM	DBLP-ACM	96.6	19.7
EM	DBLP-GoogleScholar	83.8	69.7
EM	Amazon-Google	63.5	42.1
EM	iTunes-Amazon	98.2	70.0
EM	Walmart-Amazon	87.0	25.5
DI	Buy	98.5	84.6
DI	Restaurant	88.4	50
ED	Hospital	97.8	23.5
ED	Adult	99.1	100*

Data Transformation

Dataset	PBE [4]	LLMPR [9]	Code Generation (Ours)
BingQL-semantics	32.0	54.0	**91.6**
BingQL-Unit	**96.0**	N/A	95.0
Stack-overflow	63.0	65.3	**87.4**
FF-GR-Trifacta	**91.0**	N/A	83.7
Head cases	**82.0**	N/A	74.6
Average	72.8	N/A	**86.46**

- **Evaluate based on rows.**
- **A single generated program can only solve a part of the dataset.**

Preliminary Results on Python Generation

Entity Matching, Data Imputation, Error Detection

Task	Dataset	LLMPR[9]	Code Generation (Ours)
EM	Fodors-Zagats	100	95.5
EM	Beer	100	75.0
EM	DBLP-ACM	96.6	19.7
EM	DBLP-GoogleScholar	83.8	69.7
EM	Amazon-Google	63.5	42.1
EM	iTunes-Amazon	98.2	70.0
EM	Walmart-Amazon	87.0	25.5
DI	Buy	98.5	84.6
DI	Restaurant	88.4	50
ED	Hospital	97.8	23.5
ED	Adult	99.1	100*

Data Transformation

Dataset	PBE [4]	LLMPR [9]	Code Generation (Ours)
BingQL-semantics	32.0	54.0	**91.6**
BingQL-Unit	**96.0**	N/A	95.0
Stack-overflow	63.0	65.3	**87.4**
FF-GR-Trifacta	**91.0**	N/A	83.7
Head cases	**82.0**	N/A	74.6
Average	72.8	N/A	**86.46**

- **Evaluate based on rows.**
- **A single generated program can only solve a part of the dataset.**

Preliminary Results on Python Generation

Entity Matching, Data Imputation, Error Detection

Task	Dataset	LLMPR[9]	Code Generation (Ours)
EM	Fodors-Zagats	100	95.5
EM	Beer	100	75.0
EM	DBLP-ACM	96.6	19.7
EM	DBLP-GoogleScholar	83.8	69.7
EM	Amazon-Google	63.5	42.1
EM	iTunes-Amazon	98.2	70.0
EM	Walmart-Amazon	87.0	25.5
DI	Buy	98.5	84.6
DI	Restaurant	88.4	50
ED	Hospital	97.8	23.5
ED	Adult	99.1	100*

76.9

Data Transformation

Dataset	PBE [4]	LLMPR [9]	Code Generation (Ours)
BingQL-semantics	32.0	54.0	**91.6**
BingQL-Unit	**96.0**	N/A	95.0
Stack-overflow	63.0	65.3	**87.4**
FF-GR-Trifacta	**91.0**	N/A	83.7
Head cases	**82.0**	N/A	74.6
Average	72.8	N/A	**86.46**

- Evaluate based on rows.
- A single generated program can only solve a part of the dataset.

Preliminary Results on Python Generation

Entity Matching, Data Imputation, Error Detection

Task	Dataset	LLMPR[9]	Code Generation (Ours)
EM	Fodors-Zagats	100	95.5
EM	Beer	100	75.0
EM	DBLP-ACM	96.6	19.7
EM	DBLP-GoogleScholar	83.8	69.7
EM	Amazon-Google	63.5	42.1
EM	iTunes-Amazon	98.2	70.0
EM	Walmart-Amazon	87.0	25.5
DI	Buy	98.5	84.6
DI	Restaurant	88.4	50
ED	Hospital	97.8	23.5
ED	Adult	99.1	100*

76.9

- Evaluate based on rows.
- A single generated program can only solve a part of the dataset.

Data Transformation

Dataset	PBE [4]	LLMPR [9]	Code Generation (Ours)
BingQL-semantics	32.0	54.0	**91.6**
BingQL-Unit	**96.0**	N/A	95.0
Stack-overflow	63.0	65.3	**87.4**
FF-GR-Trifacta	**91.0**	N/A	83.7
Head cases	**82.0**	N/A	74.6
Average	72.8	N/A	**86.46**

- Evaluate based on tasks.

Preliminary Results on Python Generation

Entity Matching, Data Imputation, Error Detection

Task	Dataset	LLMPR[9]	Code Generation (Ours)
EM	Fodors-Zagats	100	95.5
EM	Beer	100	75.0
EM	DBLP-ACM	96.6	19.7
EM	DBLP-GoogleScholar	83.8	69.7
EM	Amazon-Google	63.5	42.1
EM	iTunes-Amazon	98.2	70.0
EM	Walmart-Amazon	87.0	25.5
DI	Buy	98.5	84.6
DI	Restaurant	88.4	50
ED	Hospital	97.8	23.5
ED	Adult	99.1	100*

76.9

- Evaluate based on rows.
- A single generated program can only solve a part of the dataset.

Data Transformation

Dataset	PBE [4]	LLMPR [9]	Code Generation (Ours)
BingQL-semantics	32.0	54.0	**91.6**
BingQL-Unit	**96.0**	N/A	95.0
Stack-overflow	63.0	65.3	**87.4**
FF-GR-Trifacta	**91.0**	N/A	83.7
Head cases	**82.0**	N/A	74.6
Average	72.8	N/A	**86.46**

- Evaluate based on tasks.
- Improved performance compare to the previous SOTA.

Preliminary Results on Python Generation

Entity Matching, Data Imputation, Error Detection

Task	Dataset	LLMPR[9]	Code Generation (Ours)
EM	Fodors-Zagats	100	95.5
EM	Beer	100	75.0
EM	DBLP-ACM	96.6	19.7
EM	DBLP-GoogleScholar	83.8	69.7
EM	Amazon-Google	63.5	42.1
EM	iTunes-Amazon	98.2	70.0
EM	Walmart-Amazon	87.0	25.5
DI	Buy	98.5	84.6
DI	Restaurant	88.4	50
ED	Hospital	97.8	23.5
ED	Adult	99.1	100*

76.9 ↓

- Evaluate based on rows.
- A single generated program can only solve a part of the dataset.

Data Transformation

Dataset	PBE [4]	LLMPR [9]	Code Generation (Ours)
BingQL-semantics	32.0	54.0	**91.6**
BingQL-Unit	**96.0**	N/A	95.0
Stack-overflow	63.0	65.3	**87.4**
FF-GR-Trifacta	**91.0**	N/A	83.7
Head cases	**82.0**	N/A	74.6
Average	72.8	N/A	**86.46**

37.6 ↑

- Evaluate based on tasks.
- Improved performance compare to the previous SOTA.

Preliminary Results on Python Generation

Entity Matching, Data Imputation, Error Detection

Task	Dataset	LLMPR[9]	Code Generation (Ours)
EM	Fodors-Zagats	100	95.5
EM	Beer	100	75.0
EM	iTunes-Amazon	96.6	19.7
EM	DBLP-GoogleScholar	83.8	69.7
EM	Amazon-Google	63.5	42.1
EM	iTunes-Amazon	98.2	70.0
EM	Walmart-Amazon	87.0	25.5
DI	Buy	98.5	84.6
DI	Restaurant	88.4	50
ED	Hospital	97.8	23.5
ED	Adult	99.1	100*

LLM API calls: 76.9

1 Million

- Evaluate based on rows.
- A single generated program can only solve a part of the dataset.

Data Transformation

Dataset	PBE [4]	LLMPR [9]	Code Generation (Ours)
BingQL-semantics	32.0	54.0	**91.6**
BingQL-Unit	**96.0**	N/A	95.0
Stack-overflow	63.0	65.3	**87.4**
FF-GR-Trifacta	**91.0**	N/A	83.7
Head cases	**82.0**	N/A	74.6
Average	72.8	N/A	**86.46**

37.6

- Evaluate based on tasks.
- Improved performance compare to the previous SOTA.

Preliminary Results on Python Generation

Entity Matching, Data Imputation, Error Detection

Task	Dataset	LLMPR[9]	Code Generation (Ours)
EM	Fodors-Zagats	100	95.5
EM	Beer	100	75.0
EM	iTunes-Amazon	96.6	19.7
EM	DBLP-GoogleScholar	83.8	69.7
EM	Amazon-Google	63.5	42.1
EM	iTunes-Amazon	98.1	70.0
EM	Walmart-Amazon	87.4	74.5
DI	Buy	98.5	84.6
DI	Restaurant	88.4	50
ED	Hospital	97.8	23.5
ED	Adult	99.1	100*

LLM API calls: 76.9

- Evaluate based on rows.
- A single generated program can only solve a part of the dataset.

Data Transformation

Dataset	PBE [4]	LLMPR [9]	Code Generation (Ours)
BingQL-semantics	32.0	54.0	**91.6**
BingQL-Unit	**96.0**	N/A	95.0
Stack-overflow	63.0	65.3	**87.4**
FF-GR-Trifacta	**91.0**	N/A	83.7
Head cases	**82.0**	N/A	74.6
Average	72.6	N/A	**86.46**

37.6

- Evaluate based on tasks.
- Improved performance compare to the previous SOTA.

1 Million -> *only* 10 calls

Preliminary Results on Python Generation

Entity Matching, Data Imputation, Error Detection

Task	Dataset	LLMPR[9]	Code Generation (Ours)
EM	Fodors-Zagats	100	95.5
EM	Beer	100	75.0
EM	[illegible]	96.6	19.7
EM	DBLP-GoogleScholar	83.8	69.7
EM	Amazon-Google	63.5	42.1
EM	iTunes-Amazo[illegible]	[illegible]	70.0
EM	Walmart-Amazon	[illegible]	[illegible]
DI	Buy	98.5	84.6
DI	Restaurant	88.4	50
ED	Hospital	97.8	23.5
ED	Adult	99.1	100*

76.9

Data Transformation

Dataset	PBE [4]	LLMPR [9]	Code Generation (Ours)
BingQL-semantics	32.0	54.0	**91.6**
BingQL-Unit	**96.0**	N/A	95.0
Stack-overflow	63.0	65.3	**87.4**
FF-GR-Trifacta	**91.0**	N/A	83.7
Head cases	**82.0**	N/A	74.6
Average	72.6	N/A	**86.46**

37.6

LLM API calls:

1 Million -> *only* 10 calls

- Evaluate based on tasks.
- Improved performance compare to the previous SOTA.

But what's the catch?

- Evaluate based on rows.
- A single generated program can only solve a part of the dataset.

We evaluate on benchmarks of these tasks:

⬇Drop: 26.2

	Phone	City
1	212/582-7200	

	Phone	City
1	212/582-7200	NYC

Data Imputation

⬇Drop: 36.7

	Hospital	Error?
1	medixal xenter	

	Hospital	Error?
1	medixal xenter	Yes

Error Detection

	Celsius	Fahrenheit
1	7	

	Celsius	Fahrenheit
1	7	44.6

Data Transformation

⬇Drop: 33.1

	Beer_Name	BV
1	IJwit IPA	6.50 %

	Beer_Name	BV
1	IJwit IPA	0.50 %

same?

No

Entity Matching

We evaluate on benchmarks of these tasks:

⬇Drop: 26.2

	Phone	City
1	212/582-7200	

⬇

	Phone	City
1	212/582-7200	NYC

Data Imputation

⬇Drop: 36.7

	Hospital	Error?
1	medixal xenter	

⬇

	Hospital	Error?
1	medixal xenter	Yes

Error Detection

⬆Gain: 13.66

	Celsius	Fahrenheit
1	7	

⬇

	Celsius	Fahrenheit
1	7	44.6

Data Transformation

⬇Drop: 33.1

	Beer_Name	BV
1	IJwit IPA	6.50 %

	Beer_Name	BV
1	IJwit IPA	0.50 %

same?

⬇

No

Entity Matching

Code Generation Framework with DuckDB SQL Macros

Data Transformation

	Bing-QL-semantics	Bing-QL-unit
GPT-4 (DuckDB-SQL)	65.3	96.0
GPT-4o (DuckDB-SQL)	67.3	**96.0**
GPT-4 (Python)	**91.6**	95.0

- **Python is good at semantics-related tasks, highly due to existing packages.**
- **Generating SQL macros can solve a good amount of unit-conversion tasks.**

Code Generation Solves Parts of the Data

```python
import re
def string_transformation(input_string):
    # Extract the numeric value from the input string
    celsius = float(re.search(r'\d+', input_string).group())
    # Convert Celsius to Fahrenheit
    fahrenheit = (celsius * 9/5) + 32
    # Format the result into the desired output string
    return f'{fahrenheit} Fahrenheit'
```

Generated code

Code Generation Solves Parts of the Data

```python
import re
def string_transformation(input_string):
    # Extract the numeric value from the input string
    celsius = float(re.search(r'\d+', input_string).group())
    # Convert Celsius to Fahrenheit
    fahrenheit = (celsius * 9/5) + 32
    # Format the result into the desired output string
    return f'{fahrenheit} Fahrenheit'
```

Generated code

	Celsius	Fahrenheit
1	7	44.6
2	1	33.8
x	…	…
n	-17.2222	1

Code Generation Solves Parts of the Data

```python
import re
def string_transformation(input_string):
    # Extract the numeric value from the input string
    celsius = float(re.search(r'\d+', input_string).group())
    # Convert Celsius to Fahrenheit
    fahrenheit = (celsius * 9/5) + 32
    # Format the result into the desired output string
    return f'{fahrenheit} Fahrenheit'
```

Generated code

	Celsius	Fahrenheit
1	7	44.6
2	1	33.8
x	…	…
n	-17.2222	1

Code Generation Solves Parts of the Data

```python
import re
def string_transformation(input_string):
    # Extract the numeric value from the input string
    celsius = float(re.search(r'\d+', input_string).group())
    # Convert Celsius to Fahrenheit
    fahrenheit = (celsius * 9/5) + 32
    # Format the result into the desired output string
    return f'{fahrenheit} Fahrenheit'
```

Generated code

	Celsius	Fahrenheit
1	7	44.6
2	1	33.8
x	…	…
n	-17.2222	1

Code Generation Solves Parts of the Data

```python
import re
def string_transformation(input_string):
    # Extract the numeric value from the input string
    celsius = float(re.search(r'\d+', input_string).group())
    # Convert Celsius to Fahrenheit
    fahrenheit = (celsius * 9/5) + 32
    # Format the result into the desired output string
    return f'{fahrenheit} Fahrenheit'
```

Generated code

	Celsius	Fahrenheit
1	7	44.6
2	1	33.8
x	…	…
n	-17.2222	1

Once generated code solution can solve one part of the problem. How are we gonna deal with the rest of the data?

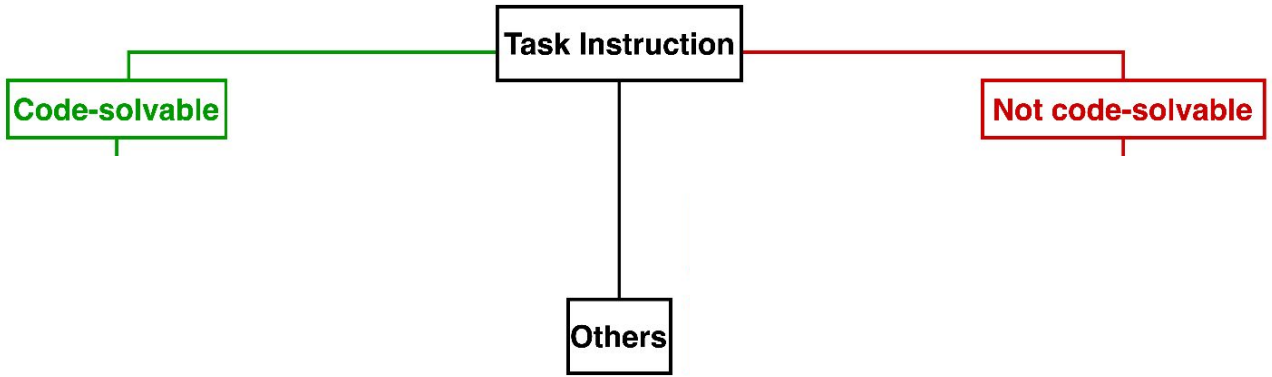
We can recursively generate code solutions for the rest of the data.

In reality, we don't have labels for the data.

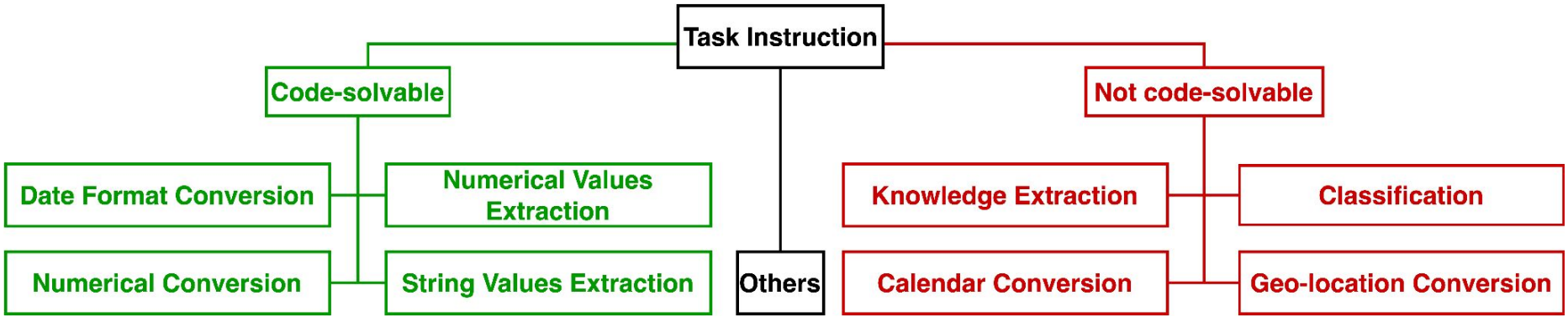
We need a data router.

In-depth Analysis of Data Transformation Tasks

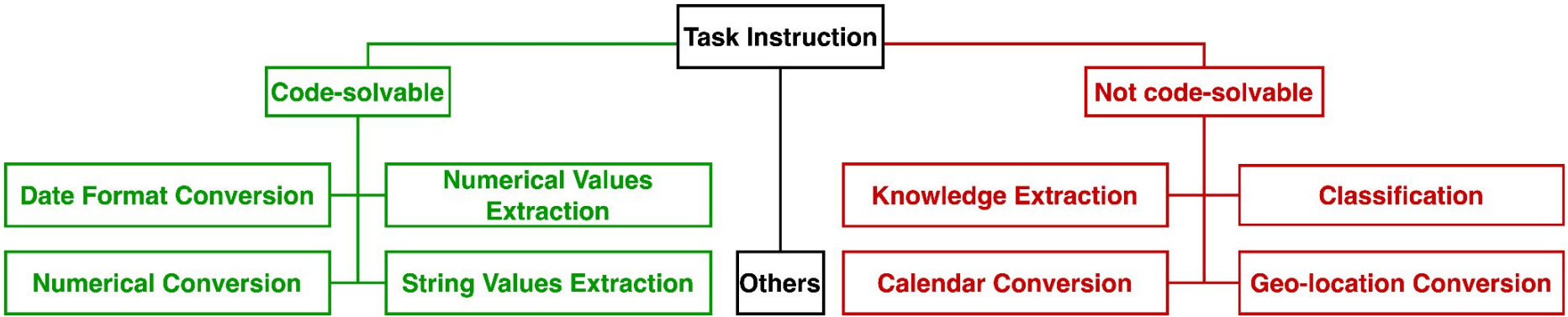
In-depth Analysis of Data Transformation Tasks



In-depth Analysis of Data Transformation Tasks



In-depth Analysis of Data Transformation Tasks



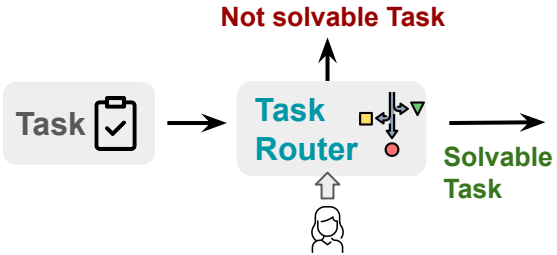
We need a task router.

Approach Overview (WIP)

Prompt used: The duck plumber
With smart tools who looks happy

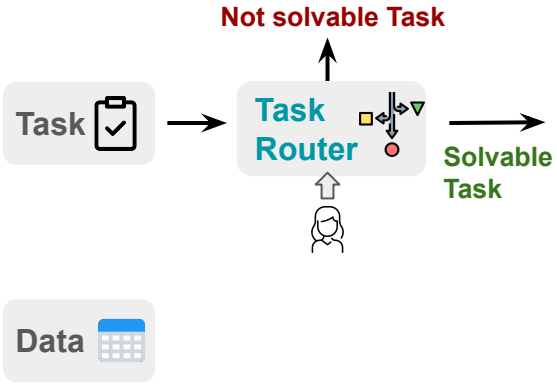
Task

Approach Overview (WIP)



Prompt used: The duck plumber
With smart tools who looks happy

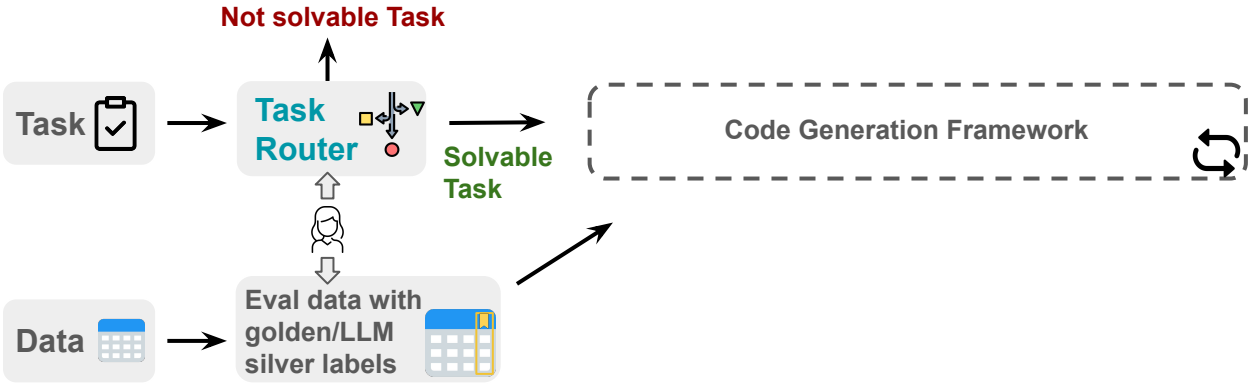
Approach Overview (WIP)



Prompt used: The duck plumber
With smart tools who looks happy

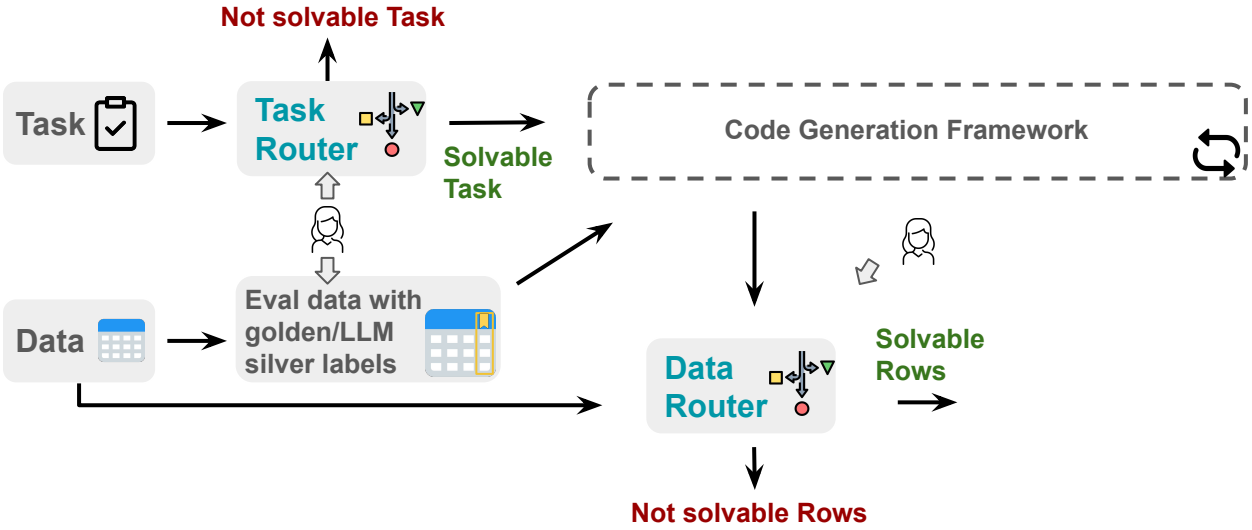
Approach Overview (WIP)

Prompt used: The duck plumber
With smart tools who looks happy



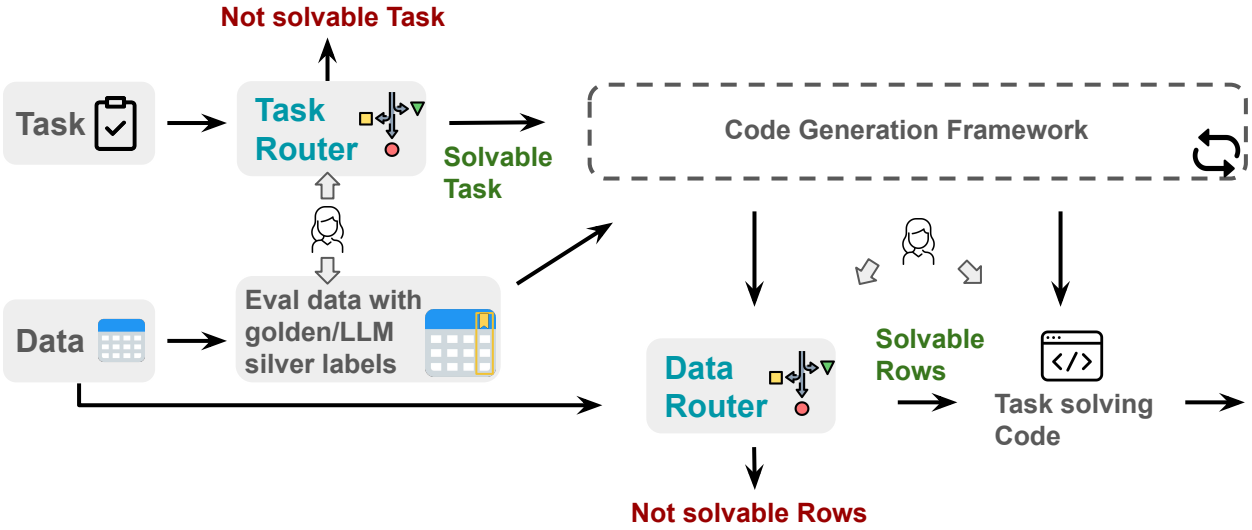
Approach Overview (WIP)

Prompt used: The duck plumber
With smart tools who looks happy



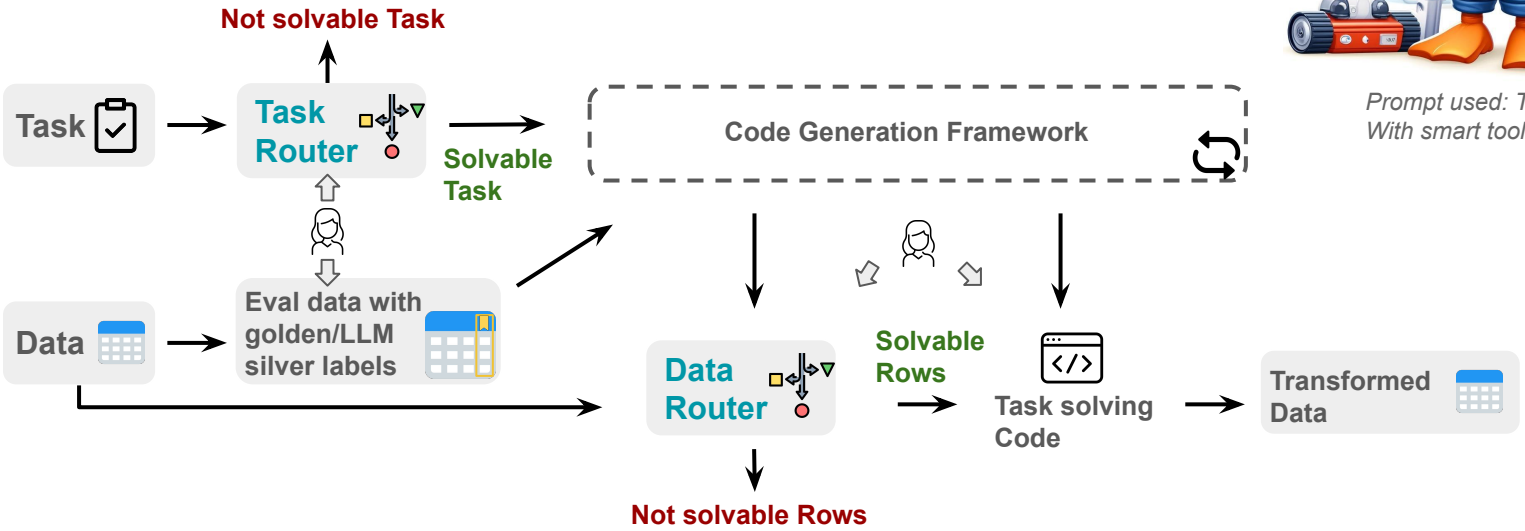
Approach Overview (WIP)

Prompt used: The duck plumber
With smart tools who looks happy



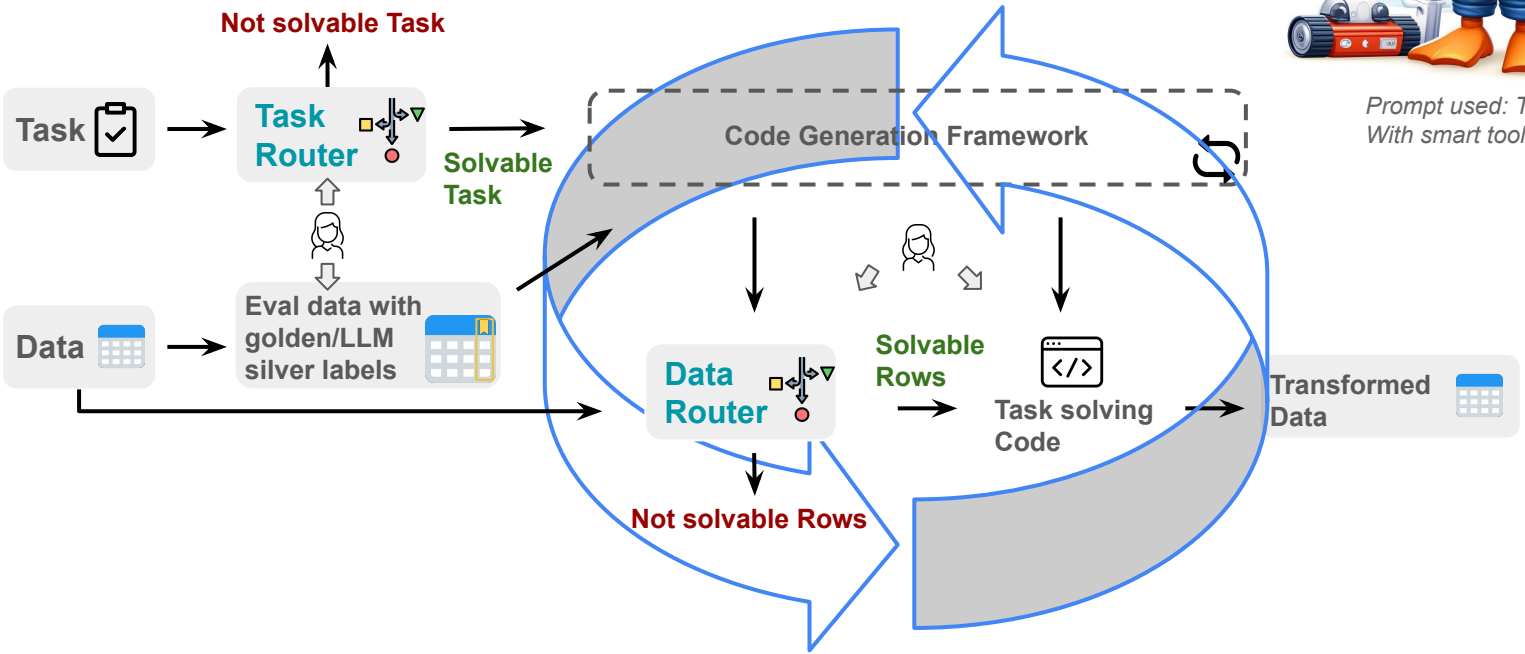
Approach Overview (WIP)

Prompt used: The duck plumber
With smart tools who looks happy



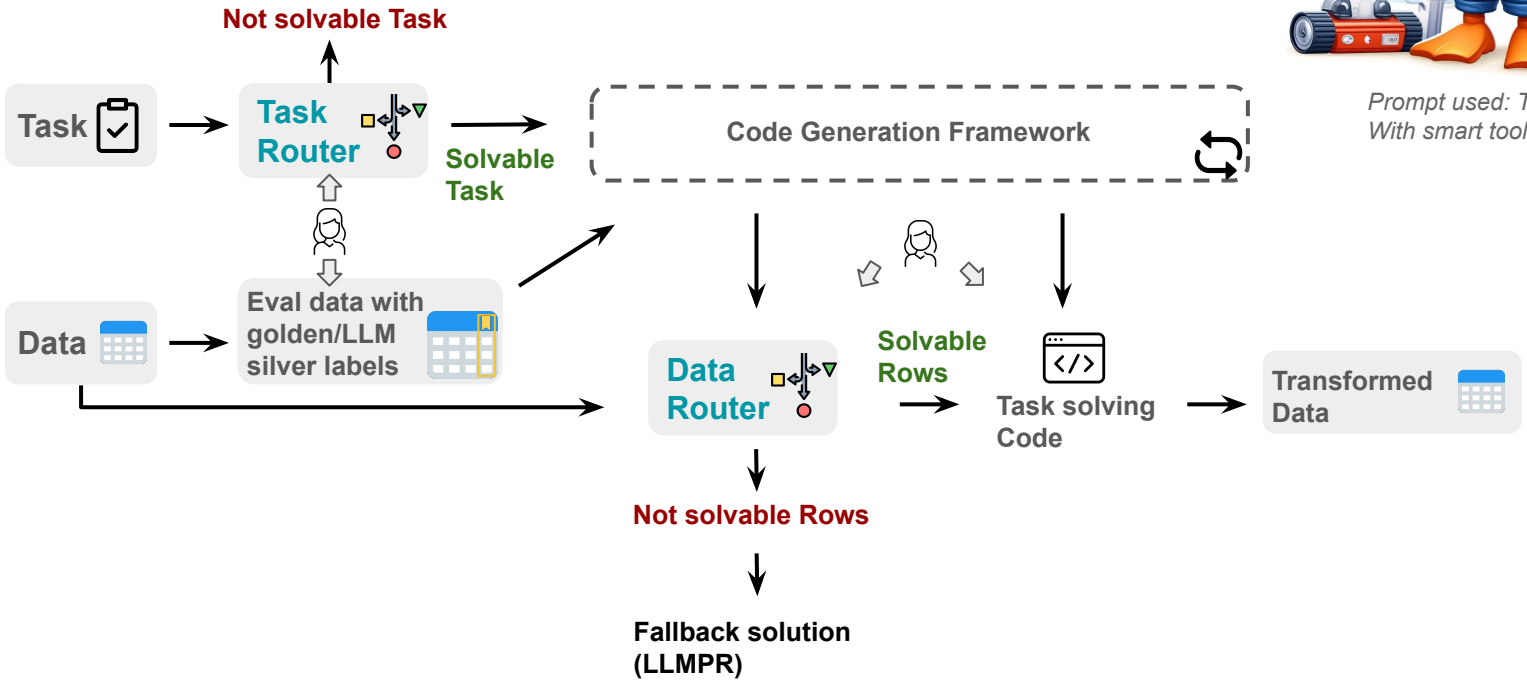
Approach Overview (WIP)

Prompt used: The duck plumber
With smart tools who looks happy



Approach Overview (WIP)

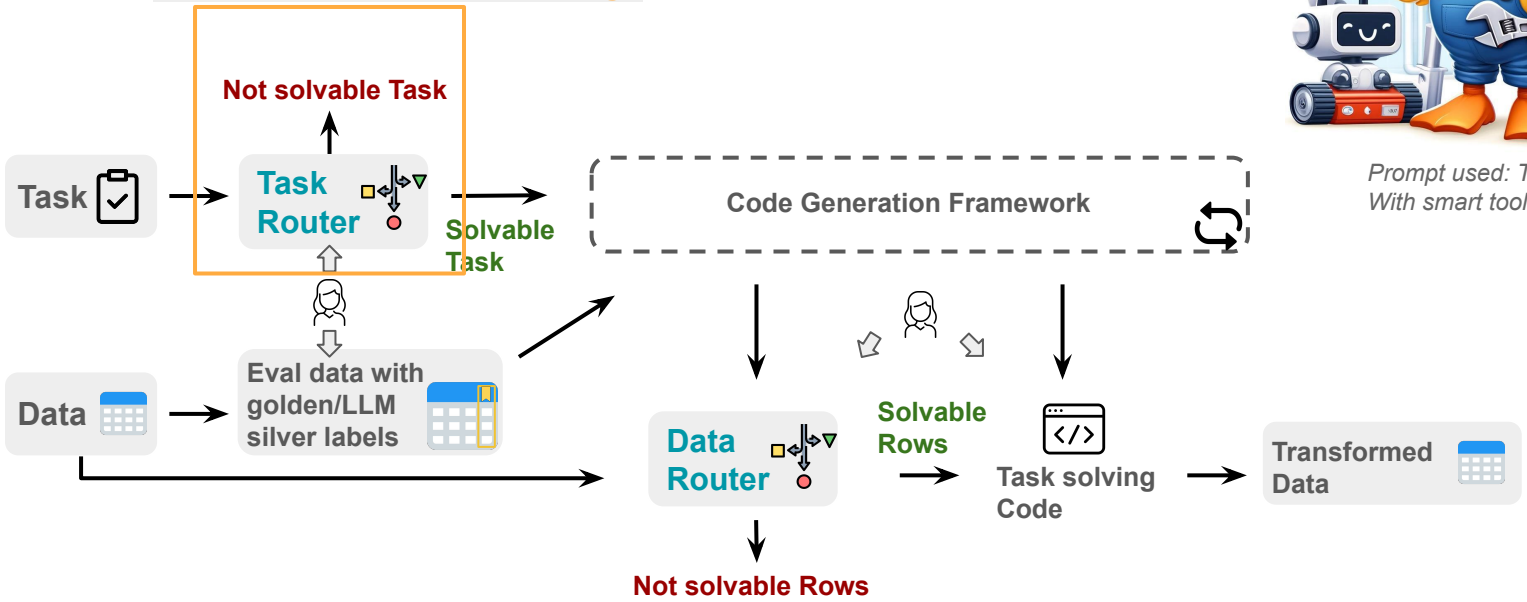
Prompt used: The duck plumber
With smart tools who looks happy



Approach Overview (WIP)

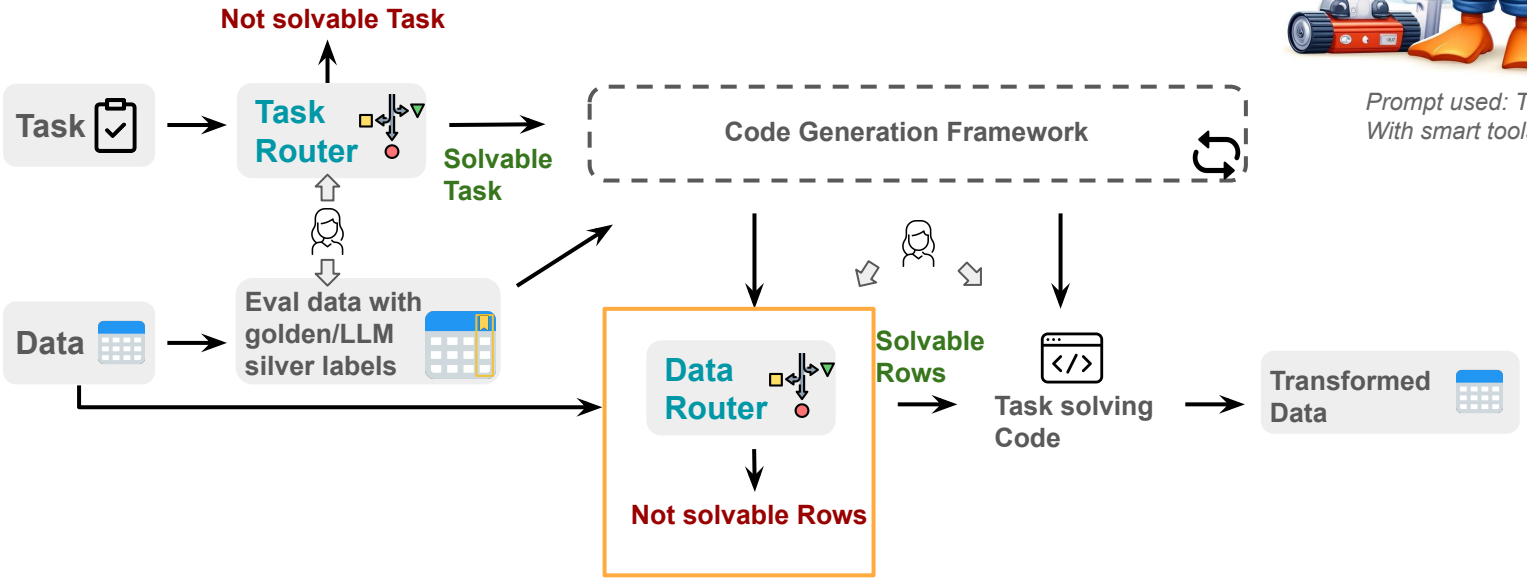
Using LLMs for mapping user's task to the taxonomy.

Prompt used: The duck plumber
With smart tools who looks happy



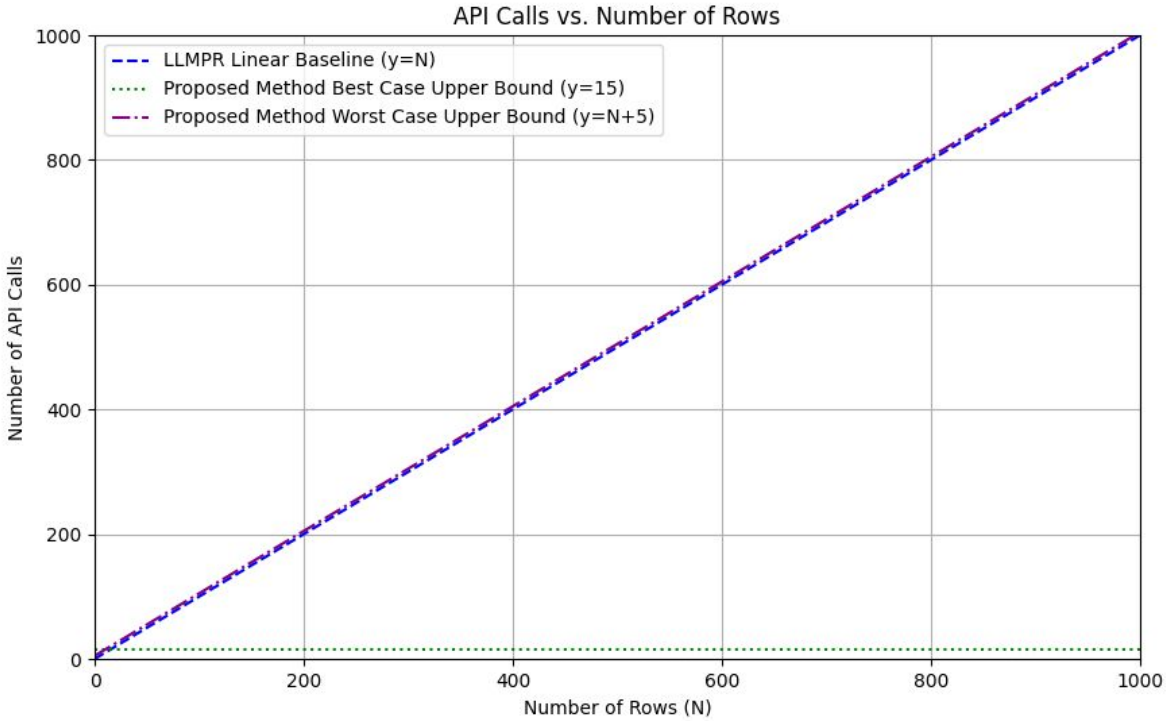
Approach Overview (WIP)

Prompt used: The duck plumber
With smart tools who looks happy



Generate input validation code (e.g. using regex) based on validated solutions.

API calls bounds

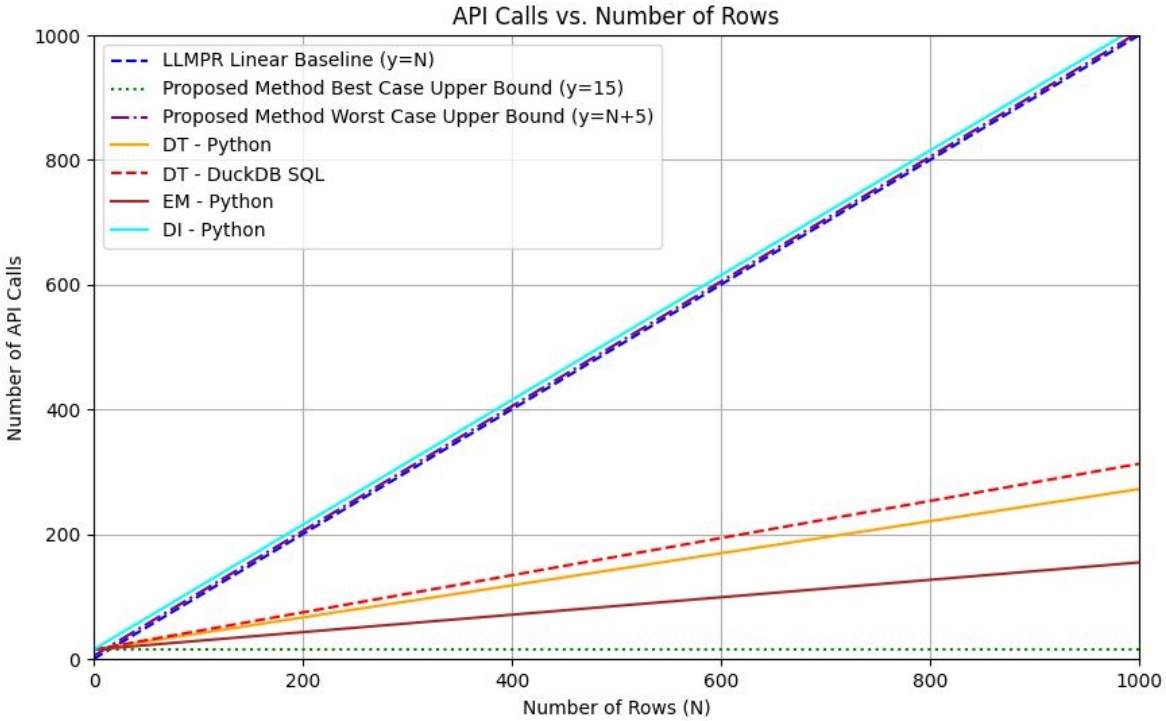


N: Number of rows
T: Number of trials = 3
R: Number of retry = 5
c : Constants for initial api calls

Baseline: APICalls(LLMPR) = N

Proposed method:
Worst case upper bound: N + c
Best case upper bound: T * R = 15

Unpublished Results



N: Number of rows
T: Number of trials = 3
R: Number of retry = 5
c : Constants for initial api calls

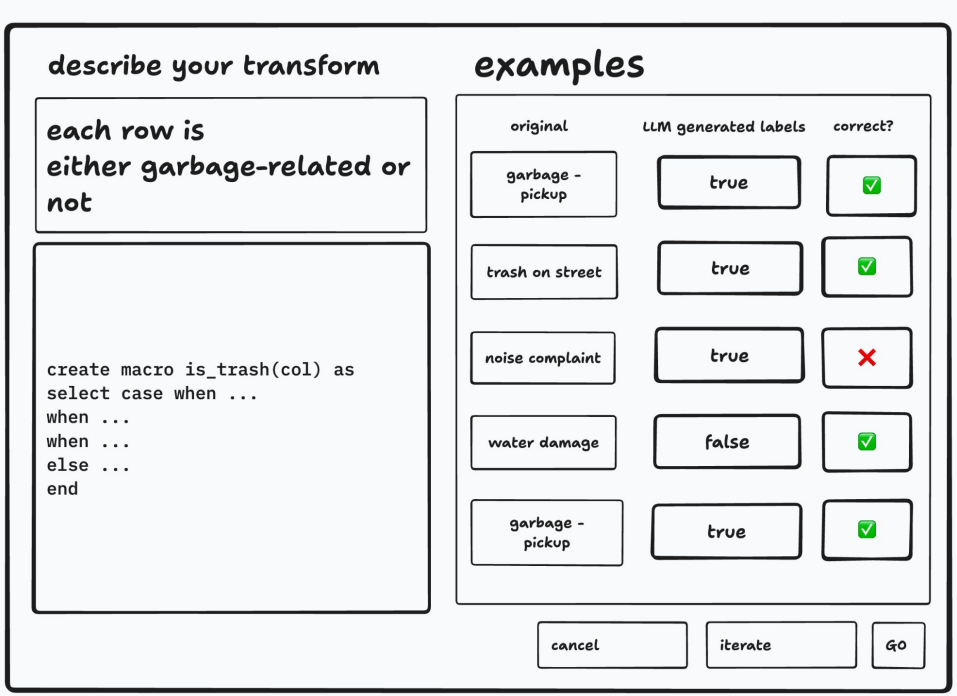
Baseline: APICalls(LLMPR) = N

Proposed method:
Worst case upper bound: N + c
Best case upper bound: T * R = 15

Source: Trust me bro

Thank you!

Next steps



Potential human-in-the-loop interface.

Cr: @Hamilton

```
PRAGMA transform('my_table',
'my_column', 'convert celsius to
fahrenheit');
```

Potential transform pragma function.