

(Why) Is My Prompt Getting Worse? Rethinking Regression Testing for Evolving LLM APIs

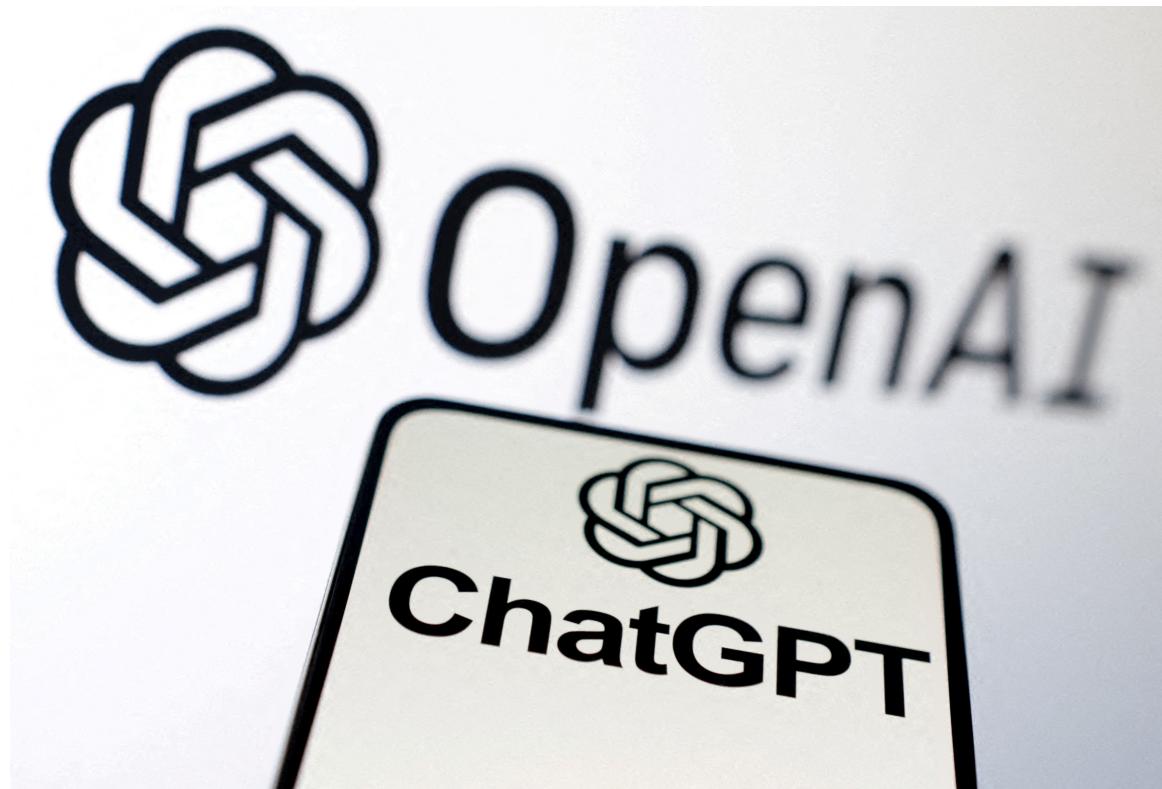
Wanqin Ma¹, **Chenyang Yang**², Christian Kästner²

The Hong Kong University of Science and Technology¹

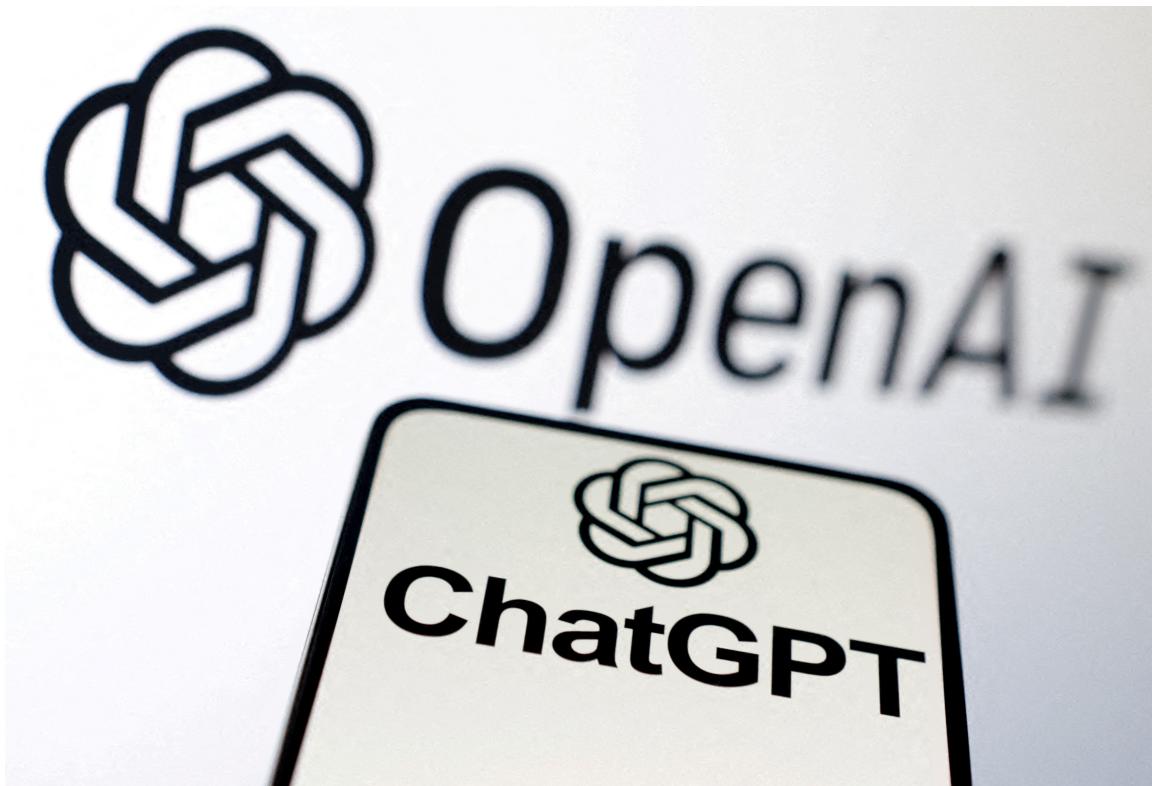
Carnegie Mellon University²



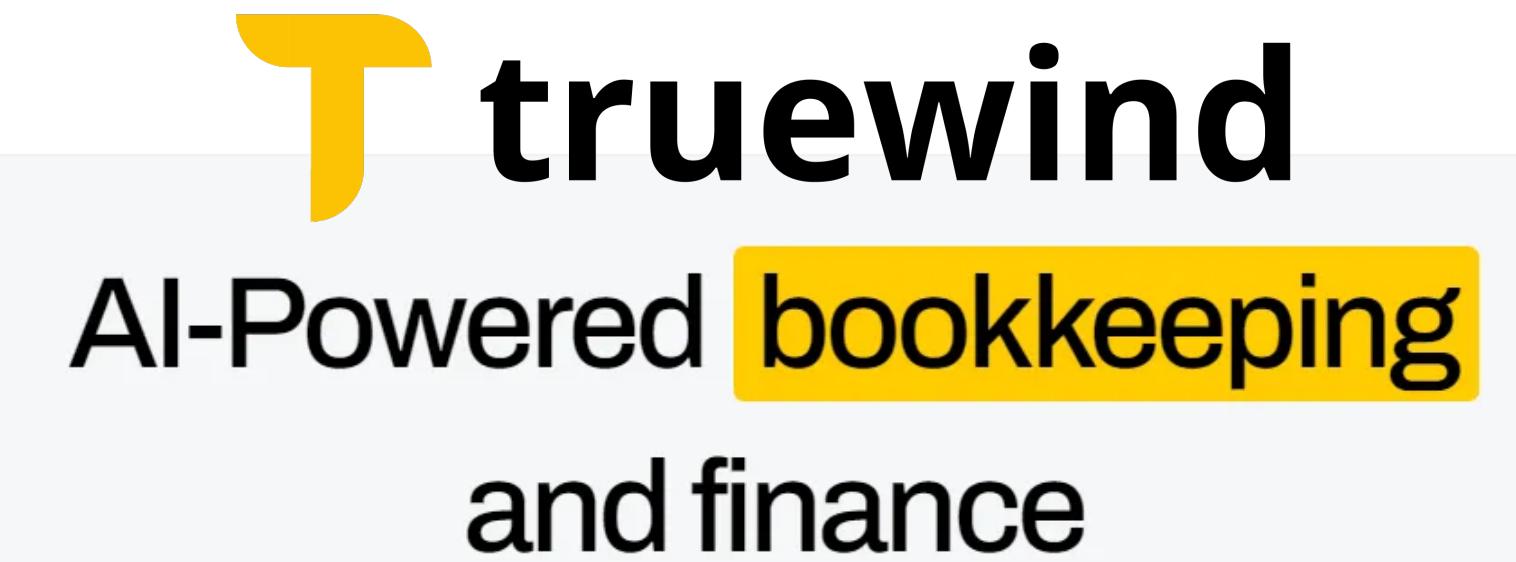
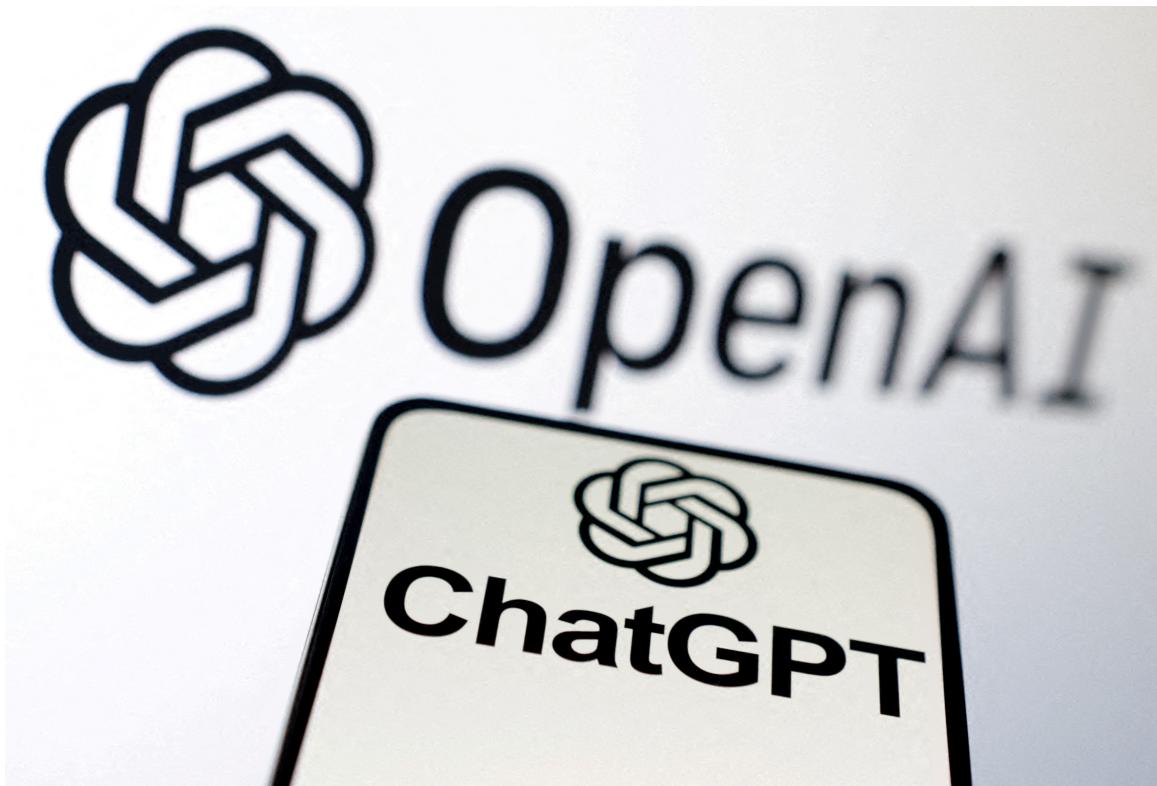
Emerging Applications Powered by LLM APIs



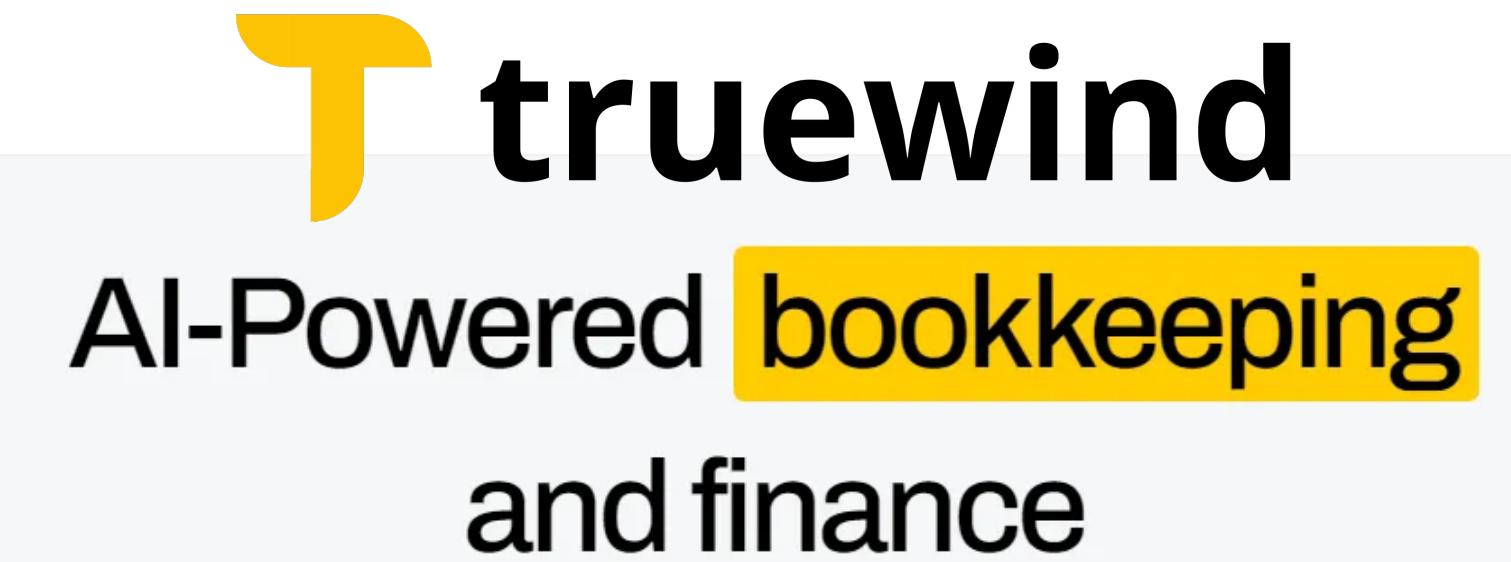
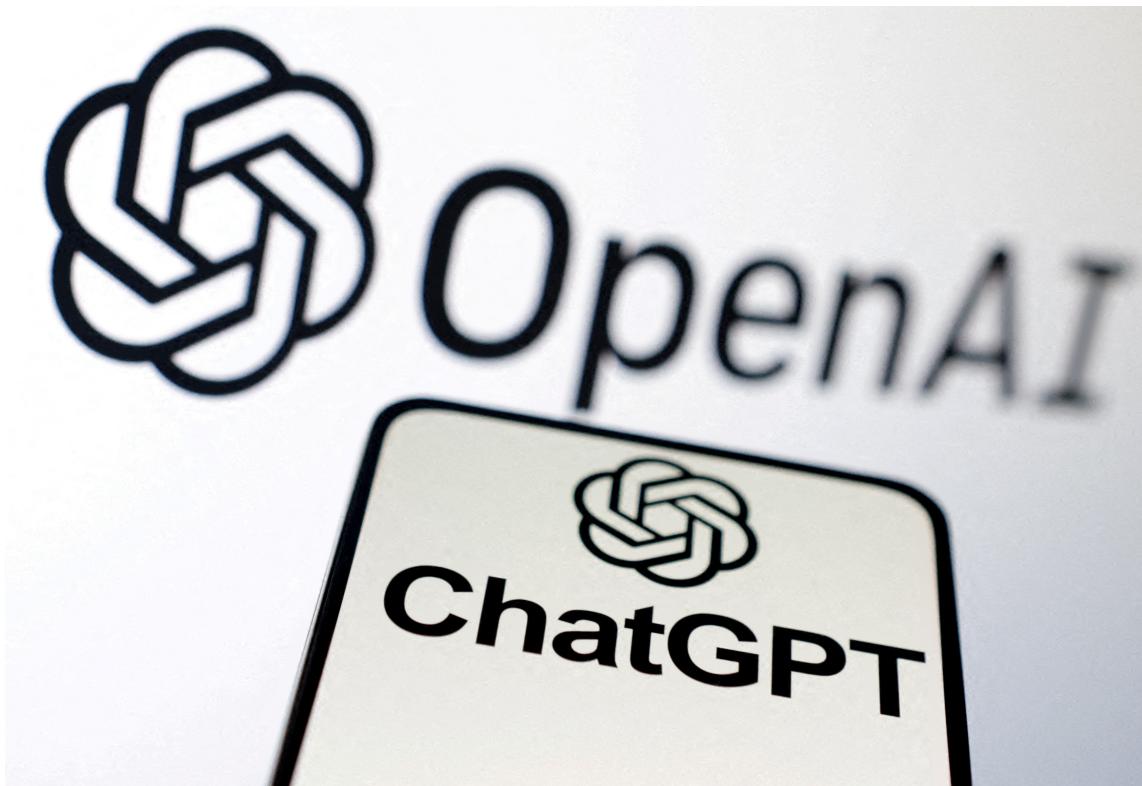
Emerging Applications Powered by LLM APIs



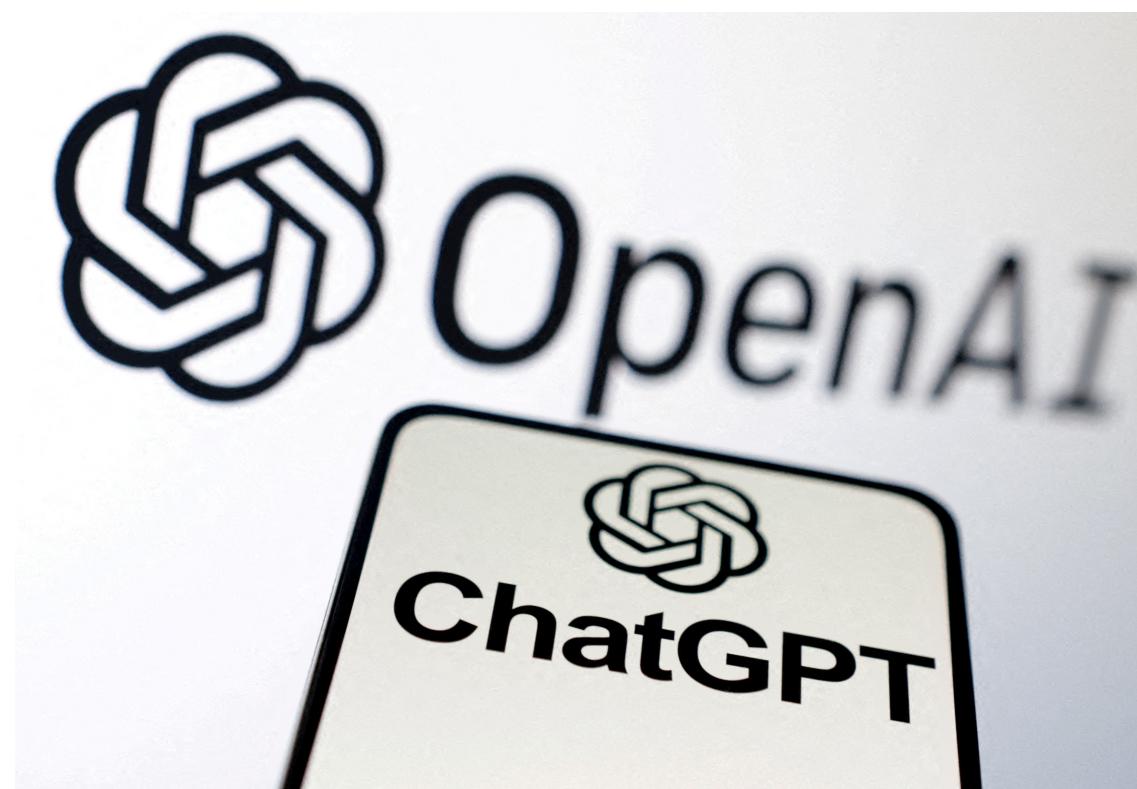
Emerging Applications Powered by LLM APIs



Emerging Applications Powered by LLM APIs

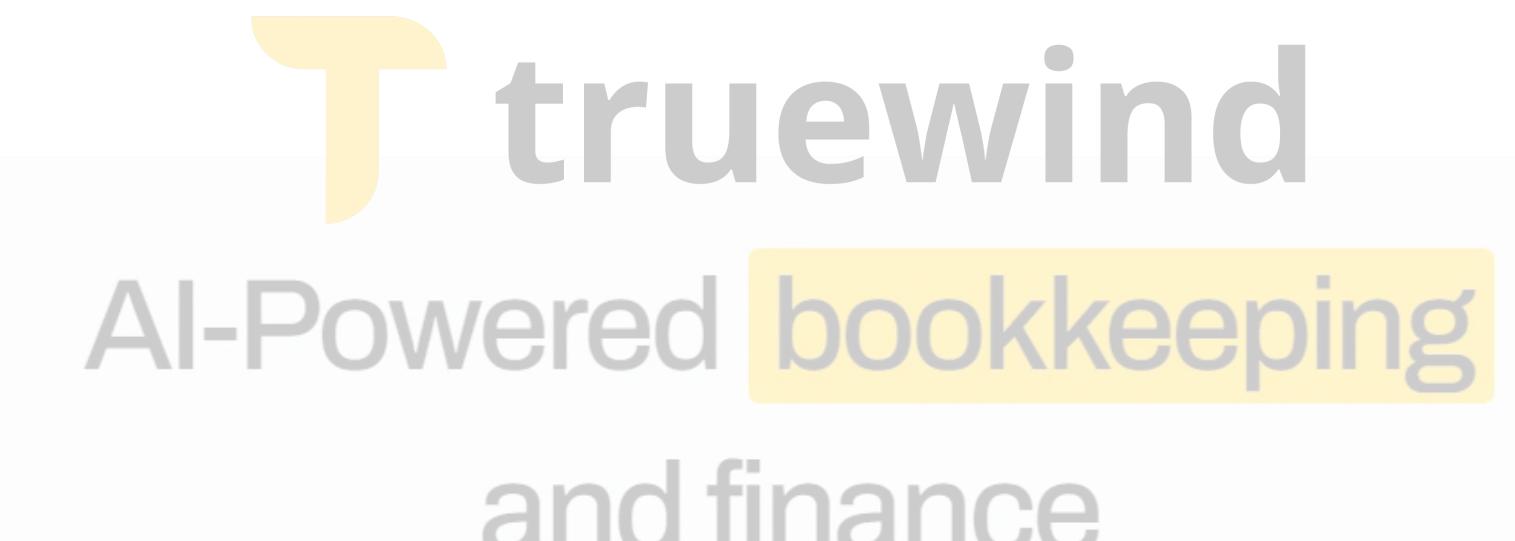


Emerging Applications Powered by LLM APIs



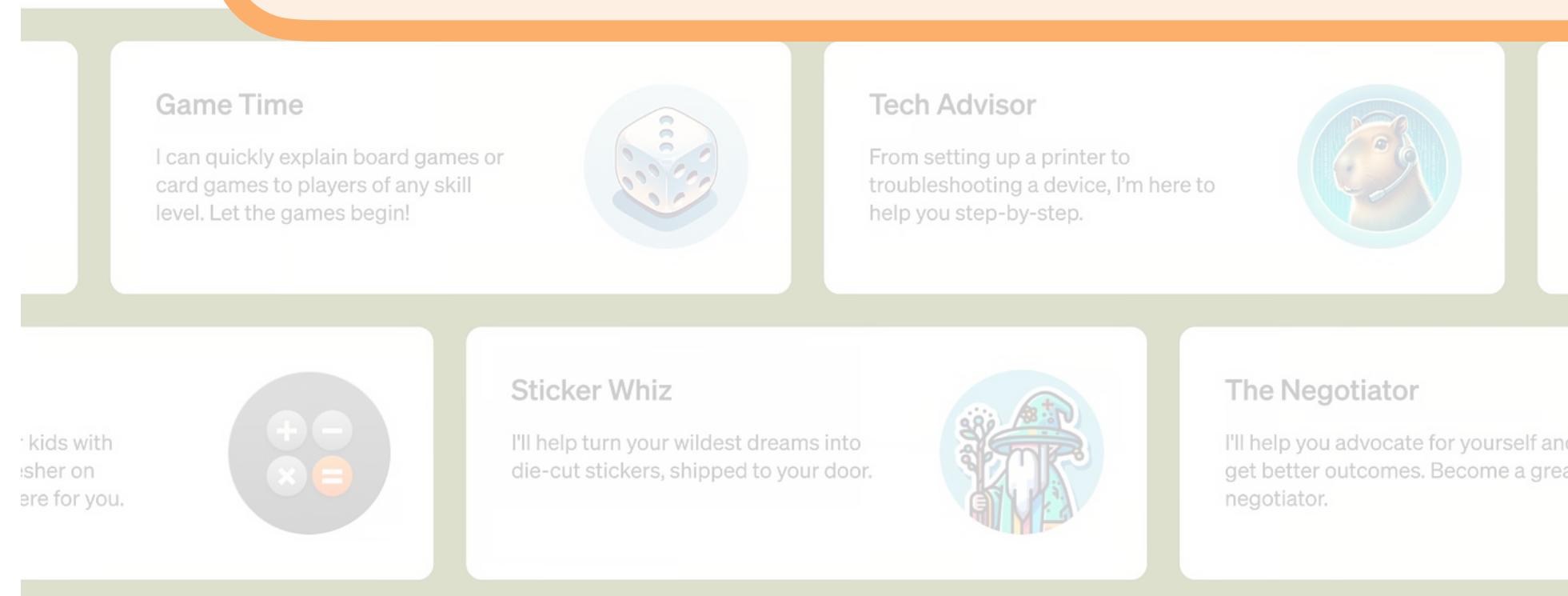
A grid of logos for various AI companies, arranged in four rows. The first row includes BerriAI (grapes), Vellum (V), MagnaPlay (person icon), Hadrius (geometric icon), Anarchy (circle icon), Zenfetch (leaf icon), PoplarML (leaf icon), and pyq (p icon). The second row includes Atri Labs (pen icon), BabylonAI (B icon), Baselit (cylinder icon), Gloo (vertical bars icon), Truewind (T icon), Chart (grid icon), Intently (grid icon), and Rollstack (R icon). The third row includes Type (t icon), Unstatiq (cube icon), UpTrain AI (circle icon), Persana AI (wavy lines icon), Baseplate (stacked bars icon), Turntable (turntable icon), Untether Labs (wavy lines icon), Magicflow (M icon), Blocktool (B icon), Extend (blue diamond icon), Squack (chicken icon), Wild Moose (moose icon), Linum (L icon), Bluebirds (bird icon), ofOne (moose icon), Buildt (circle icon), JustPaid (J icon), Layup (L icon), and Rubbrband (R icon). The fourth row includes Vocode (V icon), Quazel (Q icon), Orchid (speech bubble icon), 222 (222 icon), Ivy (spiral icon), AlphaWatch (watch icon), Chima (chima icon), and Pair AI (pair icon). In the center of the grid is a yellow callout box with the text "The AI Boom in Y Combinator".

Emerging Applications Powered by LLM APIs



Risk/Uncertainty from the provider side: LLM APIs are regularly updated (and deprecated)

Unexpected impact on downstream applications!



LLM APIs are Regularly Updated (and Deprecated)

GPT 4		GPT 3.5		
8k Tokens	32k Tokens	4k Tokens	16k Tokens	Instruct
 GPT 4	 GPT 3.5			
8k Tokens Open to those who have previously paid any bills	32k Tokens Invite only	4k Tokens Generally available	16k Tokens New Generally available	Instruct TBA
gpt-4 Stable name. Always points to the latest release.	gpt-4-32k Stable name. Always points to the latest release.	gpt-3.5-turbo Stable name. Always points to the latest release.	gpt-3.5-turbo-16k Stable name. Always points to the latest release.	gpt-3.5-turbo-instruct "Instruct style" model, not fine-tuning model. Replaces text-davinci-003 .
gpt-4-0613 Latest release. Functions added.	gpt-4-32k-0613 Latest release. Functions added. Better comprehension of larger texts.	gpt-3.5-turbo-0613 Latest release. Functions added. 25% price reduction for input tokens.	gpt-3.5-turbo-16k-0613 Latest release. Functions added. 25% price reduction for input tokens.	
Shutdown TBA	Shutdown TBA	Shutdown TBA	Shutdown TBA	Shutdown TBA
gpt-4-0314 Shuts down 13 June '24 New	gpt-4-32k-0314 Shuts down 13 June '24 New	gpt-3.5-turbo-0301 Shuts down 13 June '24 New		

LLM APIs are Regularly Updated (and Deprecated)

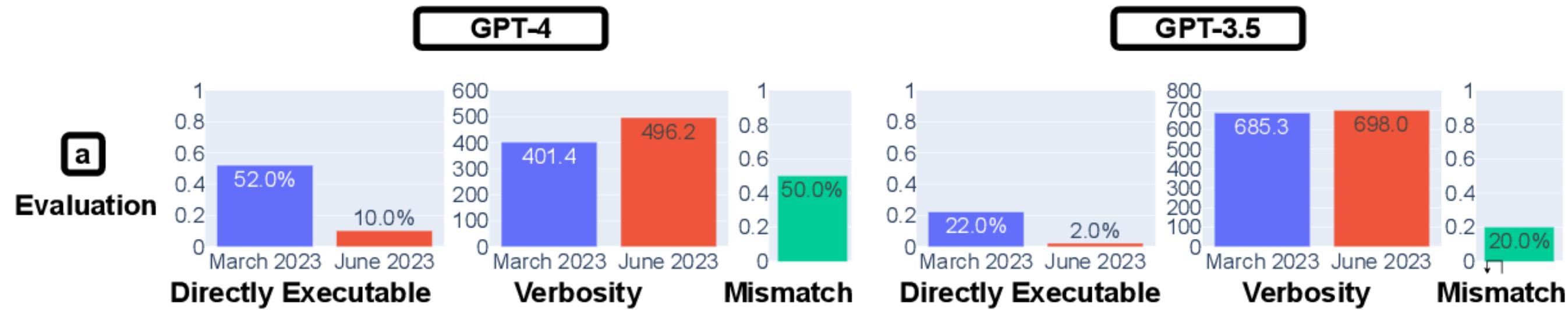
GPT 4		GPT 3.5		
8k Tokens	32k Tokens	4k Tokens	16k Tokens	Instruct
 GPT 4	 GPT 3.5			
8k Tokens Open to those who have previously paid any bills	32k Tokens Invite only	4k Tokens Generally available	16k Tokens New Generally available	Instruct TBA
gpt-4 Stable name. Always points to the latest release.	gpt-4-32k Stable name. Always points to the latest release.	gpt-3.5-turbo Stable name. Always points to the latest release.	gpt-3.5-turbo-16k Stable name. Always points to the latest release.	gpt-3.5-turbo-instruct "Instruct style" model, not fine-tuning model. Replaces text-davinci-003 .
gpt-4-0613 Latest release. Functions added.	gpt-4-32k-0613 Latest release. Functions added. Better comprehension of larger texts.	gpt-3.5-turbo-0613 Latest release. Functions added. 25% price reduction for input tokens.	gpt-3.5-turbo-16k-0613 Latest release. Functions added. 25% price reduction for input tokens.	
Shutdown TBA	Shutdown TBA	Shutdown TBA	Shutdown TBA	Shutdown TBA
gpt-4-0314 Shuts down 13 June '24 New	gpt-4-32k-0314 Shuts down 13 June '24 New	gpt-3.5-turbo-0301 Shuts down 13 June '24 New		

LLM APIs are Regularly Updated (and Deprecated)

GPT 4		GPT 3.5		
8k Tokens	32k Tokens	4k Tokens	16k Tokens	Instruct
 GPT 4	 GPT 3.5			
8k Tokens Open to those who have previously paid any bills	32k Tokens Invite only	4k Tokens Generally available	16k Tokens New Generally available	Instruct TBA
gpt-4 Stable name. Always points to the latest release.	gpt-4-32k Stable name. Always points to the latest release.	gpt-3.5-turbo Stable name. Always points to the latest release.	gpt-3.5-turbo-16k Stable name. Always points to the latest release.	gpt-3.5-turbo-instruct "Instruct style" model, not fine-tuning model. Replaces text-davinci-003 .
gpt-4-0613 Latest release. Functions added.	gpt-4-32k-0613 Latest release. Functions added. Better comprehension of larger texts.	gpt-3.5-turbo-0613 Latest release. Functions added. 25% price reduction for input tokens.	gpt-3.5-turbo-16k-0613 Latest release. Functions added. 25% price reduction for input tokens.	
Shutdown TBA	Shutdown TBA	Shutdown TBA	Shutdown TBA	Shutdown TBA
gpt-4-0314 Shuts down 13 June '24 New	gpt-4-32k-0314 Shuts down 13 June '24 New	gpt-3.5-turbo-0301 Shuts down 13 June '24 New		

Forced changes

LLM API Updates Affect Applications (unexpectedly)



a
Evaluation

b
Example

March 2023

```
class Solution(object):
    def isFascinating(self, n):
        concatenated_number = str(n) +str(2 * n) + str(3 * n)
        return sorted(concatenated_number)==['1', '2', '3', '4', '5', '6', '7', '8', '9']
```

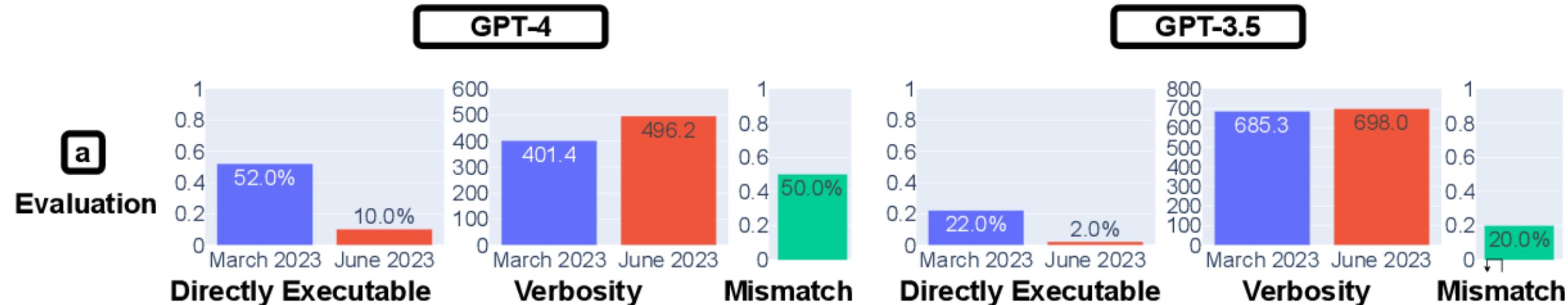
June 2023

```
```python
class Solution(object):
 def isFascinating(self, n):
 # Concatenate n, 2*n and 3*n
 s = str(n) + str(n*2) + str(n*3)
 # Check if the length of s is 9 and contains all digits from 1 to 9
 return len(s) == 9 and set(s) == set('123456789')
```
...
```

```
class Solution(object):
    def isFascinating(self, n):
        # Convert n to string for easier manipulation
        n_str = str(n)
        # Concatenate n, 2*n, and 3*n
        concatenated_str = n_str + str(2*n) + str(3*n)
        # Check if concatenated string contains all digits from 1 to 9
        # exactly once
        if set(concatenated_str) == set('123456789'):
            return True
        else:
            return False
```

```
```python
class Solution(object):
 def isFascinating(self, n):
 # Convert n to string
 n_str = str(n)
 # Concatenate n with 2*n and 3*n
 concatenated_str = n_str + str(2 * n) + str(3 * n)
 # Check if the concatenated string contains
 # all digits from 1 to 9 exactly once
 if set(concatenated_str) == set('123456789'):
 return True
 else:
 return False
```
...
```

LLM API Updates Affect Applications (unexpectedly)



a
Evaluation
[Description]: You modification, the Concatenate n with 'Concatenating' type.
[Examples]:
[Note]: Solve it by class Solution:
...

b
Example

Code generation: New APIs follow formatting differently

March 2023

```
class Solution(object):
    def isFascinating(self, n):
        concatenated_number = str(n) +str(2 * n) + str(3 * n)
        return sorted(concatenated_number)=[1, '2', '3', '4', '5', '6', '7', '8', '9']
```

```
def isFascinating(self, n):
    # Convert n to string for easier manipulation
    n_str = str(n)
    # Concatenate n, 2*n, and 3*n
    concatenated_str = n_str + str(2*n) + str(3*n)
    # Check if concatenated string contains all digits from 1 to 9
    # exactly once
    if set(concatenated_str) == set('123456789'):
        return True
    else:
        return False
```

June 2023

```
```python
class Solution(object):
 def isFascinating(self, n):
 # Concatenate n, 2*n and 3*n
 s = str(n) + str(n*2) + str(n*3)
 # Check if the length of s is 9 and contains all digits from 1 to 9
 return len(s) == 9 and set(s) == set('123456789')
...```

```

```
```python
class Solution(object):
    def isFascinating(self, n):
        # Convert n to string
        n_str = str(n)
        # Concatenate n with 2*n and 3*n
        concatenated_str = n_str + str(2 * n) + str(3 * n)
        # Check if the concatenated string contains
        # all digits from 1 to 9 exactly once
        if set(concatenated_str) == set('123456789'):
            return True
        else:
            return False
...```

```

LLM API Updates Affect Applications (unexpectedly)



a
Evaluation

[Description]: You modification, the Concatenate n with 'Concatenating' to [Examples]:
[Note]: Solve it by class Solution:
...

b
Example

Code generation: New APIs follow formatting differently

March 2023

```
class Solution(object):
    def isFascinating(self, n):
        concatenated_number = str(n) +str(2 * n) + str(3 * n)
        return sorted(concatenated_number)==['1', '2', '3', '4', '5', '6', '7', '8', '9']
```

```
def isFascinating(self, n):
    # Convert n to string for easier manipulation
    n_str = str(n)
    # Concatenate n, 2*n, and 3*n
    concatenated_str = n_str + str(2*n) + str(3*n)
    # Check if concatenated string contains all digits from 1 to 9
    # exactly once
    if set(concatenated_str) == set('123456789'):
        return True
    else:
        return False
```

June 2023

```
```python
class Solution(object):
 def isFascinating(self, n):
 # Concatenate n, 2*n and 3*n
 s = str(n) + str(n*2) + str(n*3)
 # Check if the length of s is 9 and contains all digits from 1 to 9
 return len(s) == 9 and set(s) == set('123456789')
```
```

```
```python
class Solution(object):
 def isFascinating(self, n):
 # Convert n to string
 n_str = str(n)
 # Concatenate n with 2*n and 3*n
 concatenated_str = n_str + str(2 * n) + str(3 * n)
 # Check if the concatenated string contains
 # all digits from 1 to 9 exactly once
 if set(concatenated_str) == set('123456789'):
 return True
 else:
 return False
```
```

Prompt A:

Classify the text as "toxic" or "non-toxic".

Document: [text omitted]
Label: toxic

Document: [text omitted]
Label: non-toxic

Document: {input}
Label:

Prompt B:

Classify the text as "toxic" or "non-toxic".

Toxic comments can be identity attacks, insulting, obscene, sexual-explicit, or threats.

Document: {input}
Label:

Input: Any fool knows Star Trek is the best albeit lacking the technology available to more recent shows. Your post is total crap.

Output:

text-davinci-003

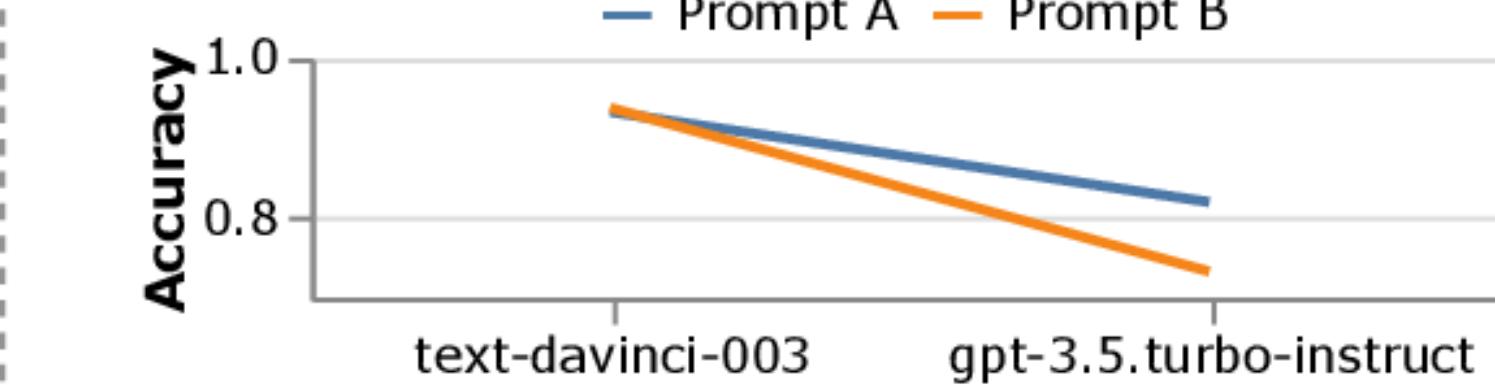
non-toxic X

toxic ✓

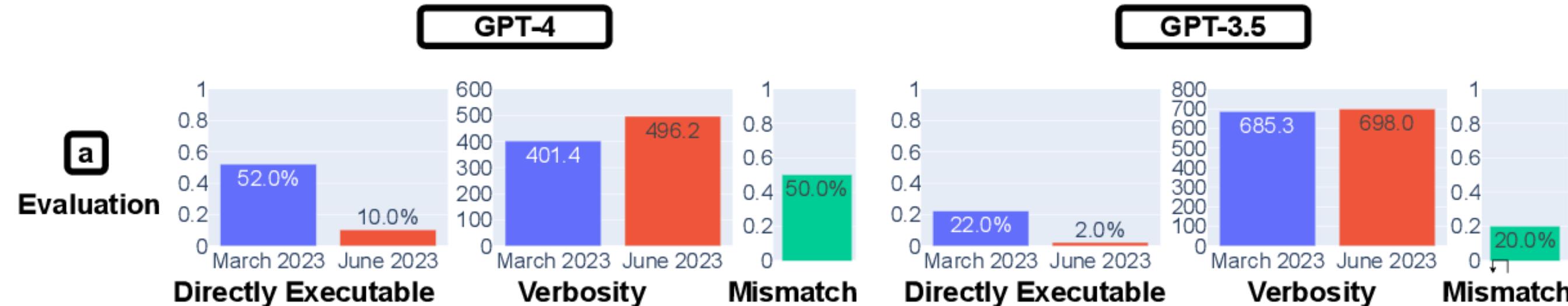
gpt-3.5-turbo-instruct

toxic ✓

non-toxic X



LLM API Updates Affect Applications (unexpectedly)



a
Evaluation

[Description]: You modification, the
Concatenate n with
'Concatenating' to
[Examples]:
[Note]: Solve it by
class Solution:
...

b
Example

Code generation: New APIs follow
formatting differently

March 2023

```
class Solution(object):
    def isFascinating(self, n):
        concatenated_number = str(n) +str(2 * n) +str(3 * n)
        return sorted(concatenated_number)==['1', '2', '3', '4', '5', '6', '7', '8', '9']
```

```
def isFascinating(self, n):
    # Convert n to string for easier manipulation
    n_str = str(n)
    # Concatenate n, 2*n, and 3*n
    concatenated_str = n_str + str(2*n) + str(3*n)
    # Check if concatenated string contains all digits from 1 to 9
    # exactly once
    if set(concatenated_str) == set('123456789'):
        return True
    else:
        return False
```

June 2023

```
```python
class Solution(object):
 def isFascinating(self, n):
 # Concatenate n, 2*n and 3*n
 s = str(n) + str(n*2) + str(n*3)
 # Check if the length of s is 9 and contains all digits from 1 to 9
 return len(s) == 9 and set(s) == set('123456789')
...
```
```

```
```python
class Solution(object):
 def isFascinating(self, n):
 # Convert n to string
 n_str = str(n)
 # Concatenate n with 2*n and 3*n
 concatenated_str = n_str + str(2 * n) + str(3 * n)
 # Check if the concatenated string contains
 # all digits from 1 to 9 exactly once
 if set(concatenated_str) == set('123456789'):
 return True
 else:
 return False
...
```
```

Prompt A:

Classify the text as "toxic"
or "non-toxic".

Document: [text omitted]
Label: toxic

Prompt B:

Classify the text as "toxic"
or "non-toxic".

Toxic comments can be
identity attacks, insulting,
obscene, sexual-explicit,

Our case study on toxicity detection:
New APIs have deteriorating performance

Input: Any fool knows Star Trek is the best albeit lacking the
technology available to more recent shows. Your post is total crap.

Output:

text-davinci-003

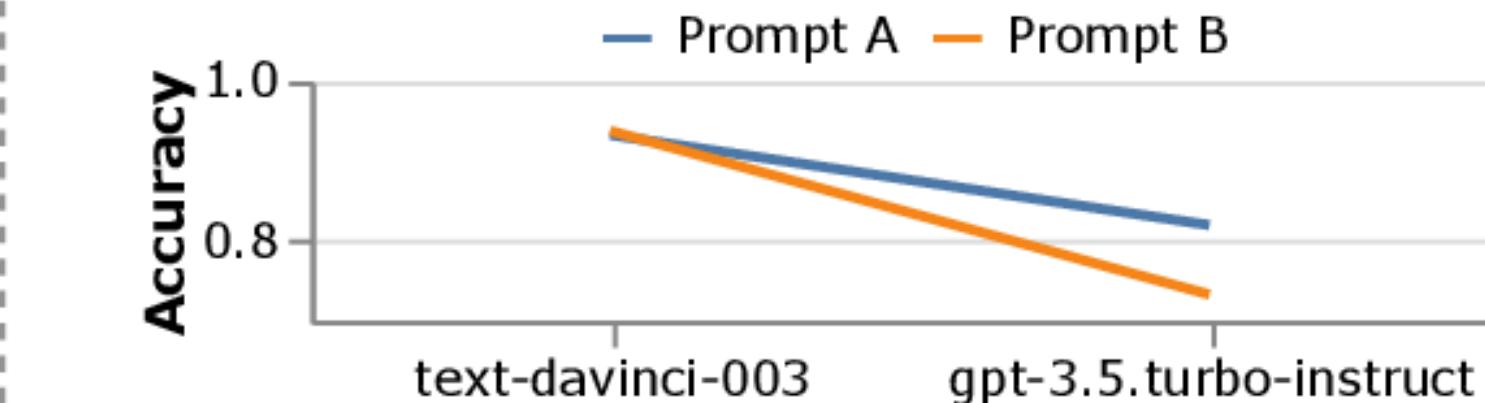
non-toxic X

toxic ✓

gpt-3.5-turbo-instruct

toxic ✓

non-toxic X



LLM API Updates Affect Applications (unexpectedly)

Experiment Setup: We study 5 APIs from GPT-3.5 family, 4 prompting strategies, and 2 datasets

Prompt A:
Classify the text as "toxic" or "non-toxic".

Document: [text omitted]
Label: toxic

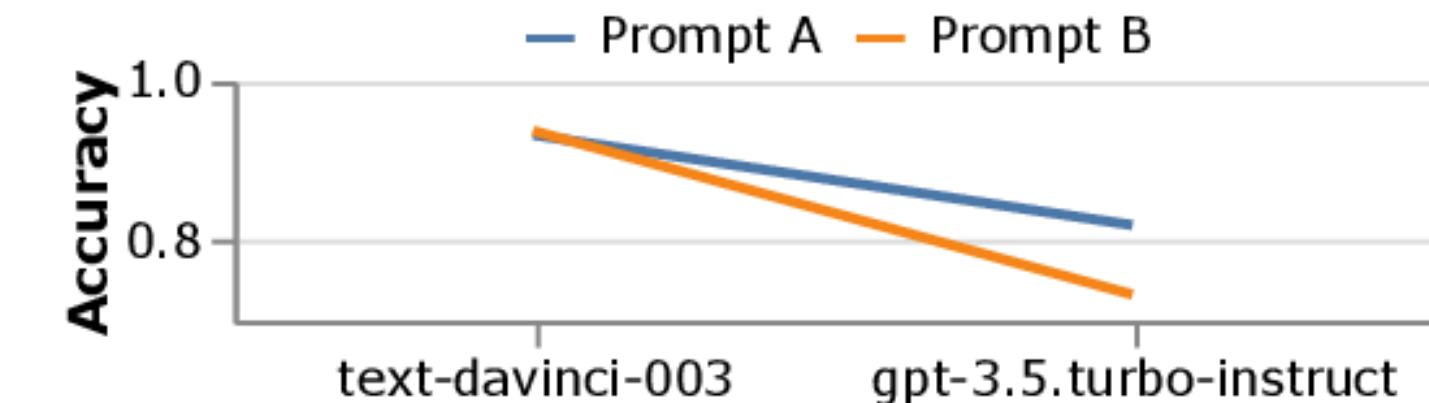
Prompt B:
Classify the text as "toxic" or "non-toxic".

Toxic comments can be identity attacks, insulting, obscene, sexual-explicit,

Our case study on toxicity detection:
New APIs have deteriorating performance

Input: Any fool knows Star Trek is the best albeit lacking the technology available to more recent shows. Your post is total crap.

| Output: | Prompt A | Prompt B |
|------------------------|-------------|-------------|
| text-davinci-003 | non-toxic ✗ | toxic ❌ |
| gpt-3.5-turbo-instruct | toxic ✗ | non-toxic ❌ |



LLM API Updates Affect Applications (unexpectedly)

Experiment Setup: We study 5 APIs from GPT-3.5 family, 4 prompting strategies, and 2 datasets

Regression is common:

58.8% prompt + model combinations drop accuracy;
10.9% individual predictions regress over updates

Prompt A:
Classify the text as "toxic" or "non-toxic".

Document: [text omitted]
Label: toxic

Prompt B:
Classify the text as "toxic" or "non-toxic".

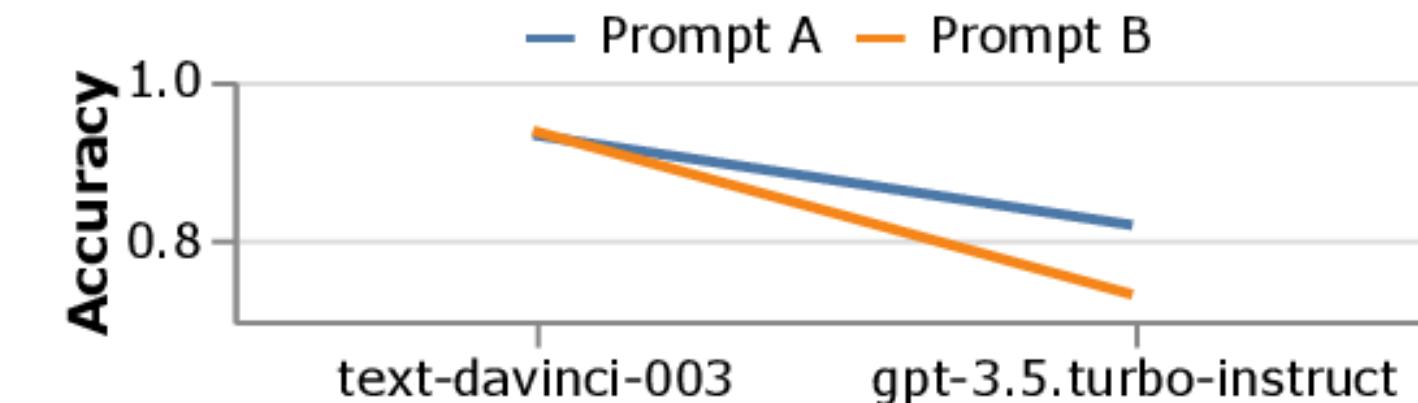
Toxic comments can be identity attacks, insulting, obscene, sexual-explicit,

Our case study on toxicity detection:

New APIs have deteriorating performance

Input: Any fool knows Star Trek is the best albeit lacking the technology available to more recent shows. Your post is total crap.

| Output: | Prompt A | Prompt B |
|------------------------|-------------|-------------|
| text-davinci-003 | non-toxic ✗ | toxic ❌ |
| gpt-3.5-turbo-instruct | toxic ✗ | non-toxic ❌ |



LLM API Updates Affect Applications (unexpectedly)

Experiment Setup: We study 5 APIs from GPT-3.5 family, 4 prompting strategies, and 2 datasets

Regression is common:

58.8% prompt + model combinations drop accuracy;
10.9% individual predictions regress over updates

Regression is not uniform across prompts:

The same update helps some prompts but hurts others

Prompt A:

Classify the text as "toxic" or "non-toxic".

Document: [text omitted]
Label: toxic

Prompt B:

Classify the text as "toxic" or "non-toxic".

Toxic comments can be identity attacks, insulting, obscene, sexual-explicit,

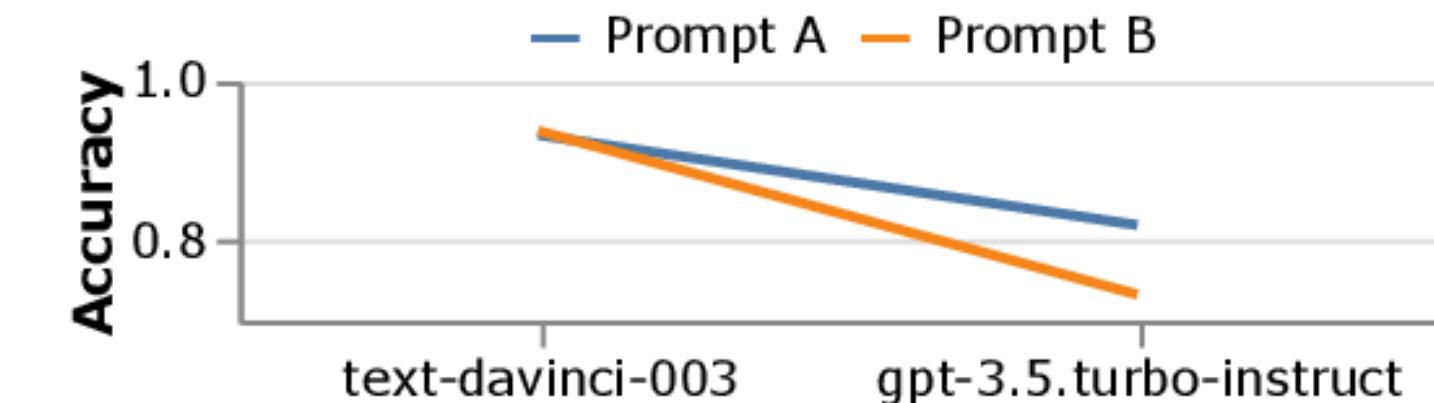
Our case study on toxicity detection:

New APIs have deteriorating performance

Input: Any fool knows Star Trek is the best albeit lacking the technology available to more recent shows. Your post is total crap.

Output:

| | Prompt A | Prompt B |
|------------------------|-------------|-------------|
| text-davinci-003 | non-toxic ✗ | toxic ❌ |
| gpt-3.5-turbo-instruct | toxic ✗ | non-toxic ❌ |



LLM API Updates Affect Applications (unexpectedly)

Experiment Setup: We study 5 APIs from GPT-3.5 family, 4 prompting strategies, and 2 datasets

Regression is common:

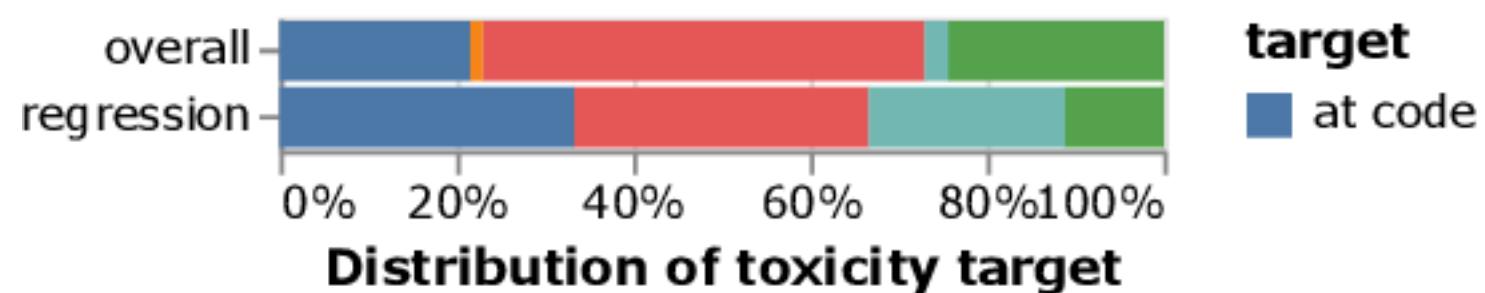
58.8% prompt + model combinations drop accuracy;
10.9% individual predictions regress over updates

Regression is not uniform across prompts:

The same update helps some prompts but hurts others

Regression is not uniform across subpopulations:

Some subpopulations are affected more than others



Prompt A:
Classify the text as "toxic" or "non-toxic".

Document: [text omitted]
Label: toxic

Prompt B:
Classify the text as "toxic" or "non-toxic".

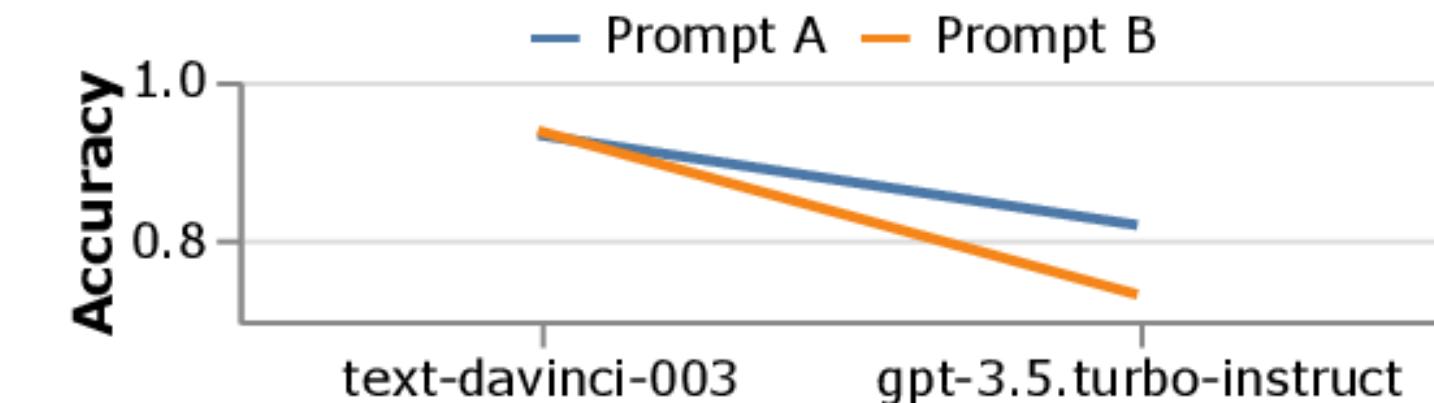
Toxic comments can be identity attacks, insulting, obscene, sexual-explicit,

Our case study on toxicity detection:

New APIs have deteriorating performance

Input: Any fool knows Star Trek is the best albeit lacking the technology available to more recent shows. Your post is total crap.

| Output: | Prompt A | Prompt B |
|------------------------|-------------|-------------|
| text-davinci-003 | non-toxic ✗ | toxic ✗ |
| gpt-3.5-turbo-instruct | toxic ✗ | non-toxic ✗ |



LLM API Updates Affect Applications (unexpectedly)

Experiment Setup: We study 5 APIs from GPT-3.5 family, 4 prompting strategies, and 2 datasets

Regression is common:

58.8% prompt + model combinations drop accuracy;

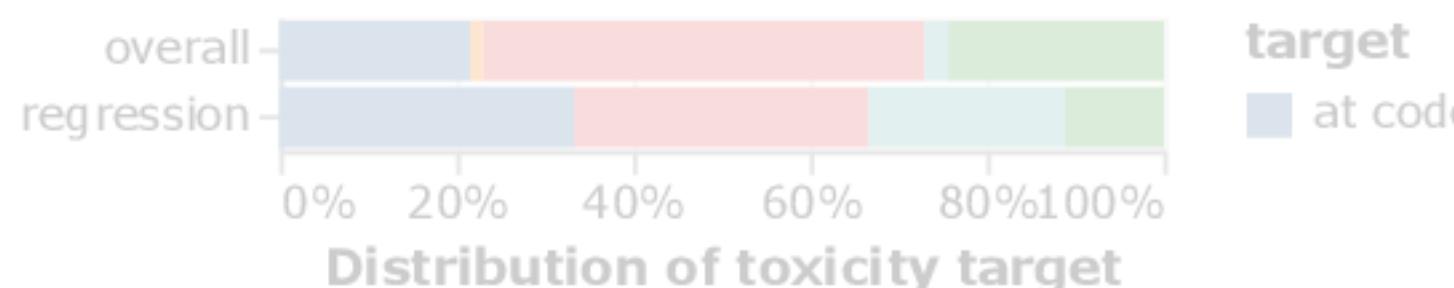
10.9%

Reg...
The

Application developers:
Is my prompt getting worse after API updates? 🤔

Regression is not uniform across subpopulations:

Some subpopulations are affected more than others



Prompt A:
Classify the text as "toxic" or "non-toxic".

Document: [text omitted]
Label: toxic

Prompt B:
Classify the text as "toxic" or "non-toxic".

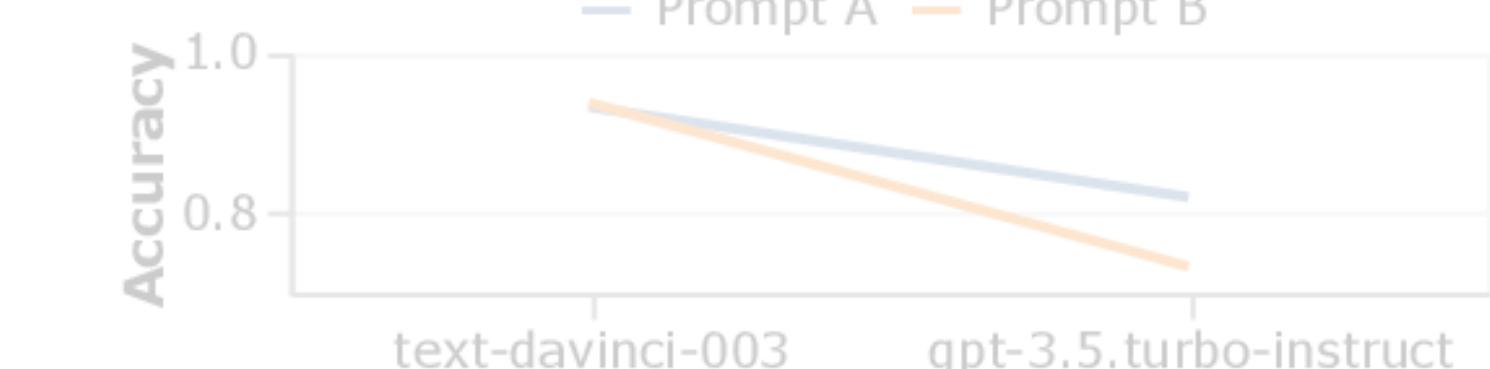
Toxic comments can be identity attacks, insulting, obscene, sexual-explicit,

Our case study on toxicity detection:

Performance

Checking the test is total crap.
Prompt B
toxic ✓

gpt-3.5-turbo-instruct toxic ✓ non-toxic ✗



Is My Prompt Getting Worse after API Updates?

Software engineers uses **regression testing** to identify changes between software versions.
We argue that this can transfer to **LLM**-based software components.

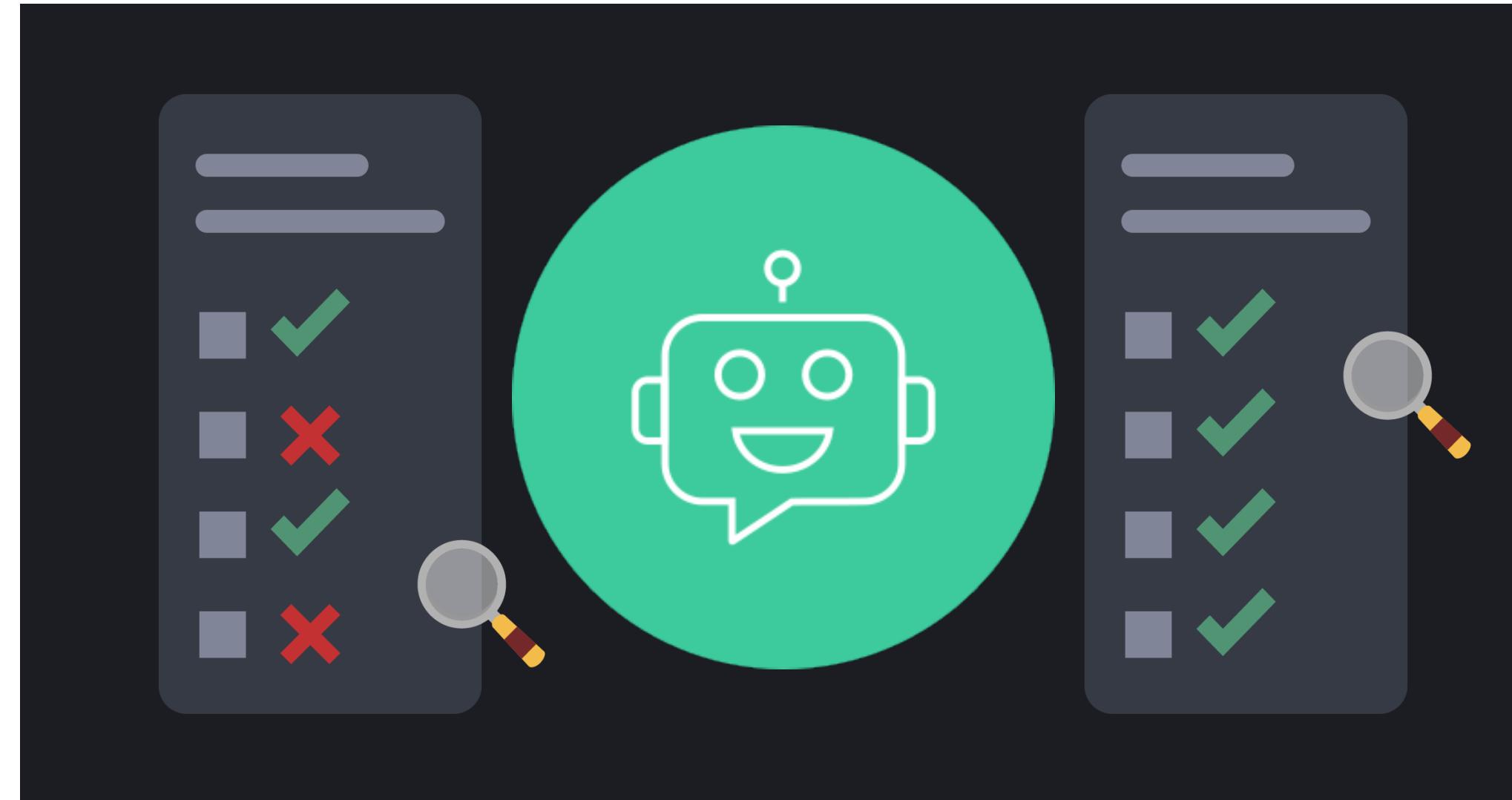


From empirical observations to research opportunities:

How should we do regression testing for **LLM**-based software components?

Is My Prompt Getting Worse after API Updates?

Software engineers uses **regression testing** to identify changes between software versions.
We argue that this can transfer to **LLM**-based software components.



From empirical observations to research opportunities:

How should we do regression testing for **LLM**-based software components?

What Counts as Regression Tests?

Traditional software: A single breaking regression test indicates a bug to fix

ML model: Overall performance drop indicates problems

What Counts as Regression Tests?

Traditional software: A single breaking regression test indicates a bug to fix

ML model: Overall performance drop indicates problems

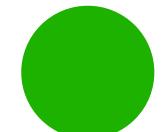
Data Points



Tweet A



Tweet B



Tweet C

Lots of data points are expected to flip after model updates.

Too fine-grained!

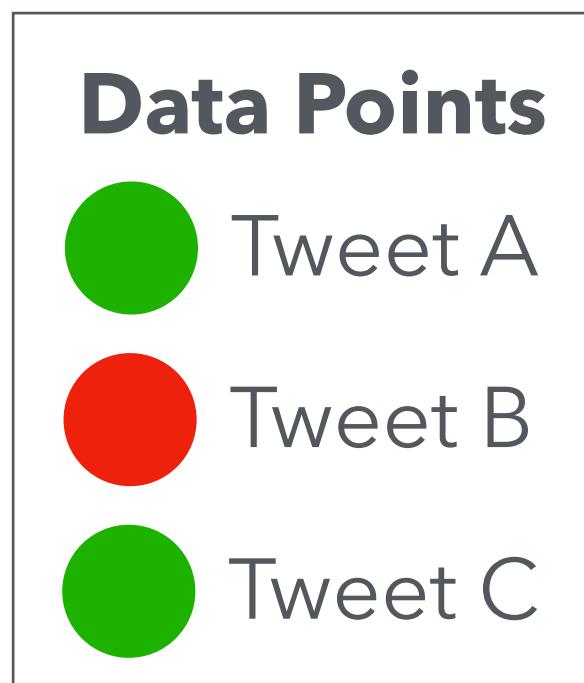
Regression is common:

10.9% individual predictions regress over updates

What Counts as Regression Tests?

Traditional software: A single breaking regression test indicates a bug to fix

ML model: Overall performance drop indicates problems

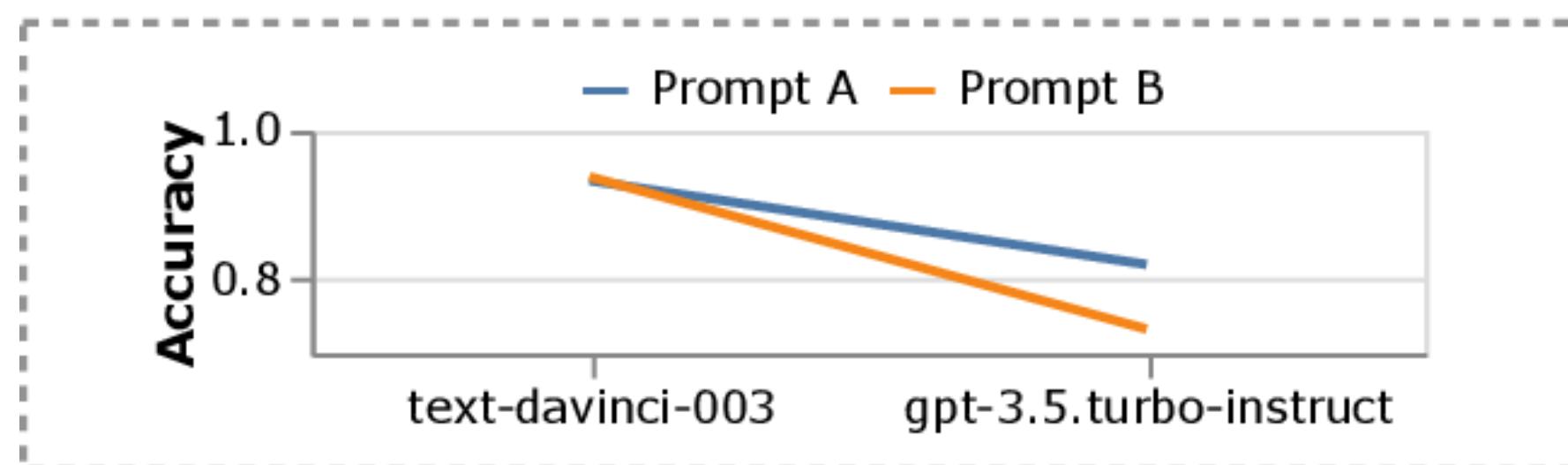


Lots of data points are expected to flip after model updates.

Too fine-grained!

Regression is common:

10.9% individual predictions regress over updates



Not helpful for understanding what leads to regressions
Possible to miss regressions on underrepresented groups

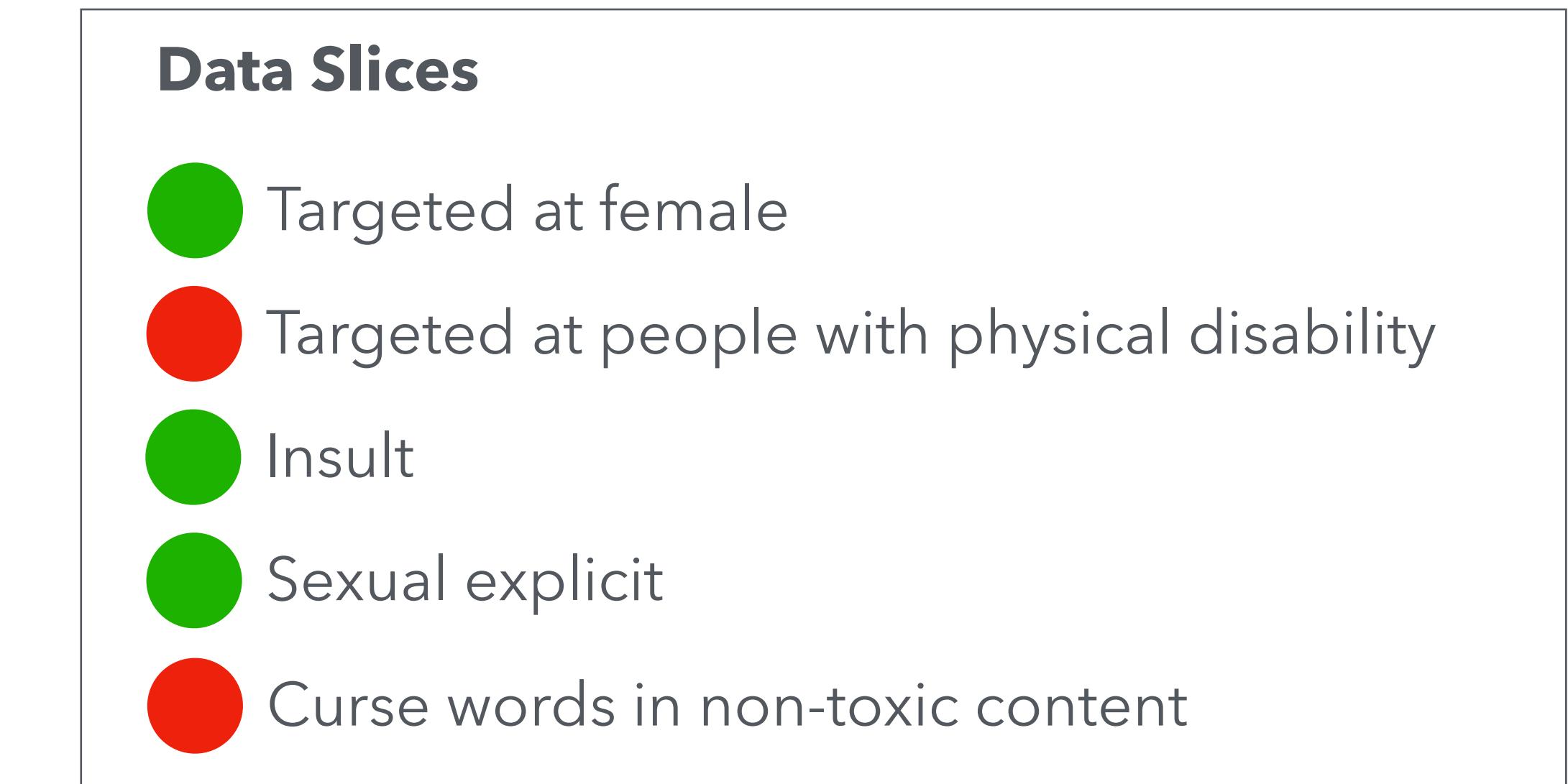
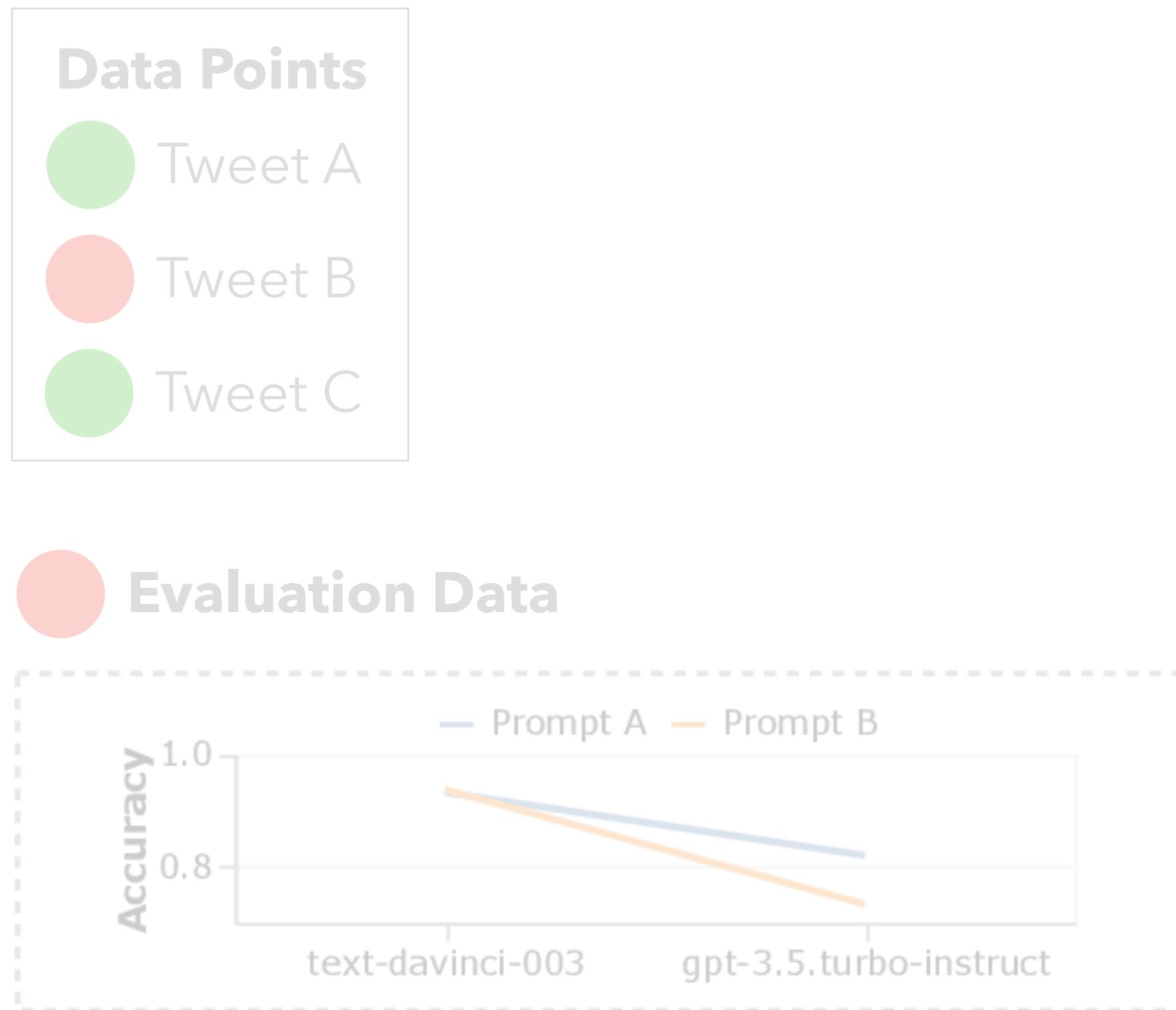
Too coarse-grained!

Regression is not uniform across subpopulations:

Some subpopulations are affected more than others

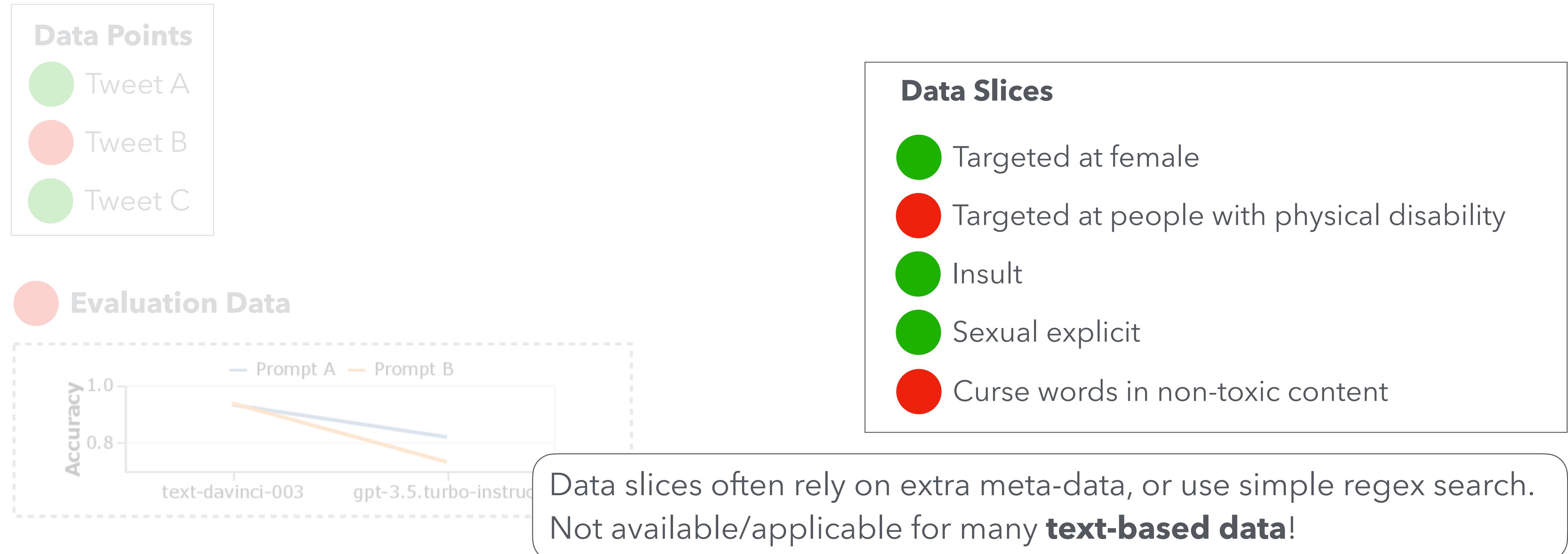
Research Opportunity #1: Test Granularity

Finding the right granularity: Use interpretable data slices to track changes



Research Opportunity #1: Test Granularity

Finding the right granularity: Use interpretable data slices to track changes



Research Opportunity #1: Test Granularity

Finding the right granularity: Use interpretable data slices to track changes

Data Points

- Tweet A
- Tweet B

Can we **scaffold** developers to **identify and collect** data slices as regression test suites?

A line graph comparing the accuracy of two prompts across different models. The y-axis is labeled 'Accuracy' and ranges from 0.8 to 1.0. The x-axis lists models: 'text-davinci-003' and 'gpt-3.5.turbo-instruc'. Two lines are plotted: 'Prompt A' (blue) and 'Prompt B' (orange). Both lines show a downward trend as the model complexity increases.

| Model | Prompt A Accuracy | Prompt B Accuracy |
|-----------------------|-------------------|-------------------|
| text-davinci-003 | ~0.95 | ~0.95 |
| gpt-3.5.turbo-instruc | ~0.82 | ~0.75 |

Data Slices

- Targeted at female
- Targeted at people with physical disability
- Insult
- Sexual explicit
- Curse words in non-toxic content

Data slices often rely on extra meta-data, or use simple regex search.
Not available/applicable for many **text-based data**!

What Do Regression Tests Track?

Regression is not uniform across prompts:

The same update helps some prompts but hurts others

Prompt A:

Classify the text as "toxic" or "non-toxic".

Document: [text omitted]
Label: toxic

Prompt B:

Classify the text as "toxic" or "non-toxic".

Toxic comments can be identity attacks, insulting, obscene, sexual-explicit,

Our case study on toxicity detection:
New APIs have deteriorating performance

Input: Any fool knows Star Trek is the best albeit lacking the technology available to more recent shows. Your post is total crap.

Output:

text-davinci-003

Prompt A non-toxic ✗

toxic ❌

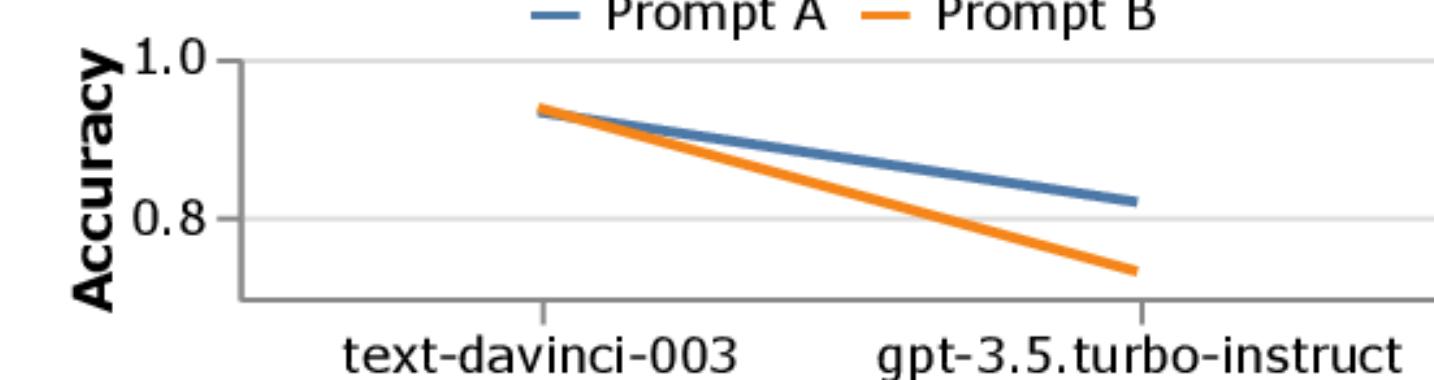
Prompt B toxic ✓

gpt-3.5-turbo-instruct

Prompt A toxic ✓

non-toxic ✗

Prompt B non-toxic ❌



What Do Regression Tests Track?

Regression is not uniform across prompts:

The same update helps some prompts but hurts others

LLM performance is known to be sensitive to prompts:
Few-shot examples vs. zero-shot instructions

Prompt A:

Classify the text as "toxic" or "non-toxic".

Document: [text omitted]
Label: toxic

Prompt B:

Classify the text as "toxic" or "non-toxic".

Toxic comments can be identity attacks, insulting, obscene, sexual-explicit,

Our case study on toxicity detection:

New APIs have deteriorating performance

Input: Any fool knows Star Trek is the best albeit lacking the technology available to more recent shows. Your post is total crap.

Output:

text-davinci-003

Prompt A non-toxic ✗

toxic ❌

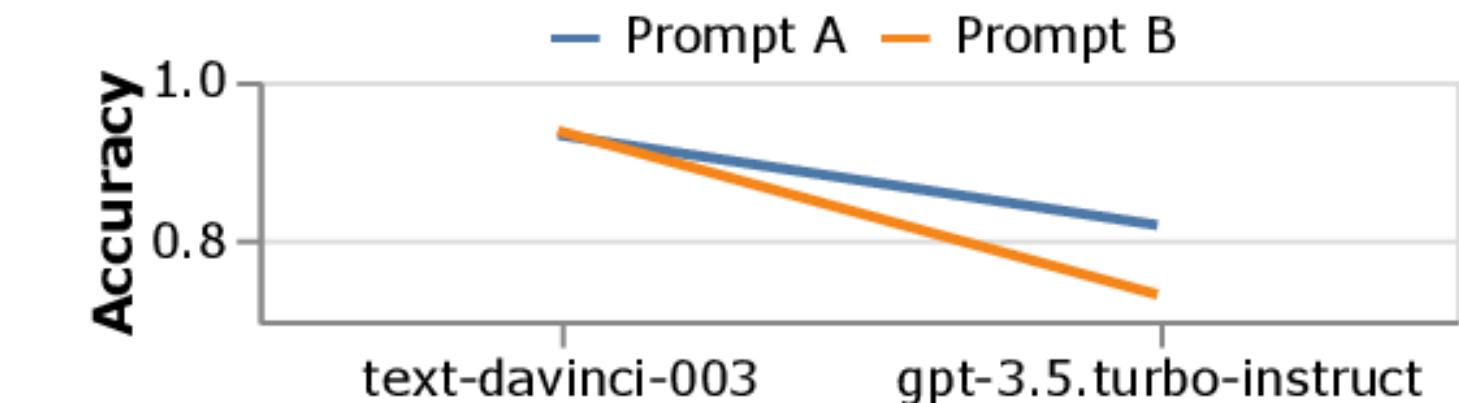
Prompt B toxic ✓

gpt-3.5-turbo-instruct

Prompt A toxic ✓

non-toxic ❌

Prompt B non-toxic ❌



Research Opportunity #2: Prompt Versioning

Regression is not uniform across prompts:

The same update helps some prompts but hurts others

LLM performance is known to be sensitive to prompts:
Few-shot examples vs. zero-shot instructions

Can we design **prompt versioning** system to help developers track behavioral changes, debug regressions, and update prompts?

A screenshot of a user interface for managing prompt versions. On the left, a sidebar lists four versions: Version 4 (30), Version 3 (30), Version 2 (44), and Version 1 (21), all updated 5 months ago. To the right, a main area shows a template for a prompt: "Summarize the following text in technical jargon." Below this is a placeholder "{content}". At the bottom, it says "Input Variables: {content}".

Reuse insights from
history prompts!

Prompt A:
Classify the text as "toxic" or "non-toxic".

Document: [text omitted]
Label: toxic

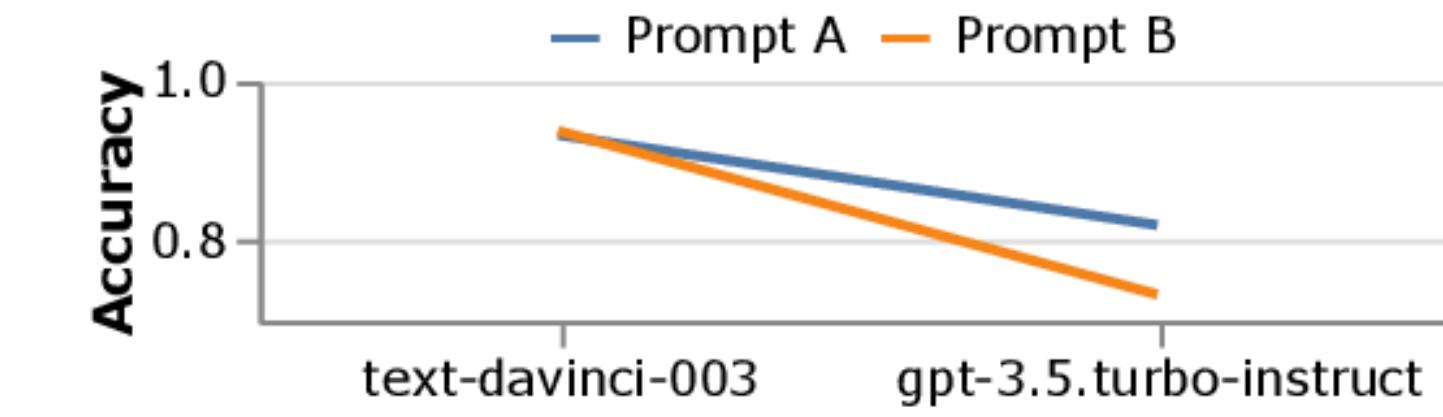
Prompt B:
Classify the text as "toxic" or "non-toxic".

Toxic comments can be identity attacks, insulting, obscene, sexual-explicit,

Our case study on toxicity detection:
New APIs have deteriorating performance

Input: Any fool knows Star Trek is the best albeit lacking the technology available to more recent shows. Your post is total crap.

| Output: | Prompt A | Prompt B |
|------------------------|-----------------|-----------------|
| text-davinci-003 | non-toxic ✗ | toxic ❌ |
| gpt-3.5-turbo-instruct | toxic ✗ | non-toxic ❌ |



How to Make Regression Tests Reliable?

LLM regression **tests** are inherently **flaky**, as LLM outputs are (often intended to be) not deterministic.

This is especially true for **generative tasks**, where there is no single ground truth.

Reproducible outputs

Beta

Chat Completions are non-deterministic by default (which means model outputs may differ from request to request). That being said, we offer some control towards deterministic outputs by giving you access to the `seed` parameter and the `system_fingerprint` response field.

<https://platform.openai.com/docs/guides/text-generation/reproducible-outputs>

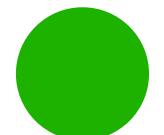
Research Opportunity #3: Non-determinism

LLM regression **tests** are inherently **flaky**, as LLM outputs are (often intended to be) not deterministic.

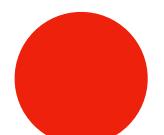
This is especially true for **generative tasks**, where there is no single ground truth.

Can we develop suitable **statistical tests** and
test minimization strategies?

Data Slices



Targeted at female



Targeted at people with physical disability

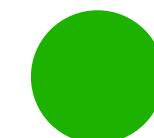
Research Opportunity #3: Non-determinism

LLM regression **tests** are inherently **flaky**, as LLM outputs are (often intended to be) not deterministic.

This is especially true for **generative tasks**, where there is no single ground truth.

Can we develop suitable **statistical tests** and **test minimization** strategies?

Data Slices



Targeted at female



Targeted at people with physical disability

Can we develop **multi-dimensional test suites** for generative tasks?

Test Criteria for Summarization



Language fluency



Truthful to source text



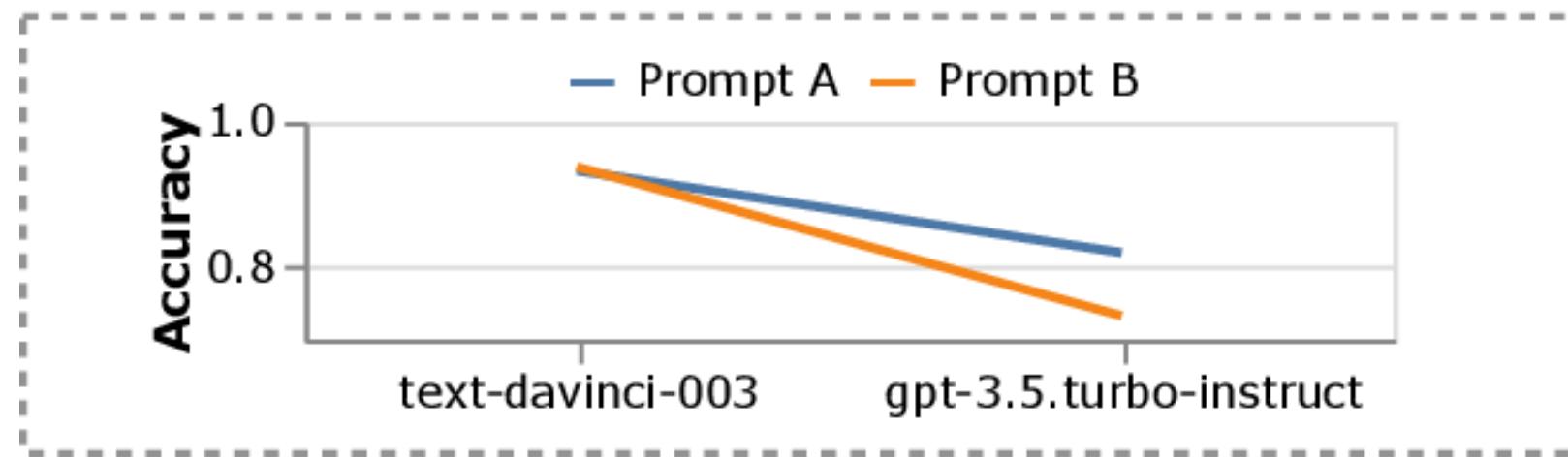
Summary conciseness



Formatting

Takeaways

LLM API updates cause regression of downstream applications



Regression is common, affects different prompting strategies differently, and is not uniform across data slices.

Lots of research opportunities for doing **regression testing for LLM-based software components**

Granularity: Can we **scaffold** developers to **identify and collect** data slices as regression test suites?

Generation: Can we develop **multi-dimensional test suites** for generative tasks?

Non-determinism: Can we develop suitable **statistical tests** and **test minimization** strategies?

Versioning: Can we design **prompt versioning** system to help developers track behavioral changes, debug regressions, and update prompts?