Neuro-Symbolic Approaches: Large Language Models + Tool Use

Tutorial on Complex Reasoning over Natural Language

Tao Yu
https://taoyds.github.io/
University of Hong Kong
07/09/2023
Language models are powerful, but they still suffer from

- Lack of interpretability
- Inconsistency
- Limited scalability
- Restricted capabilities
- ...

…
About this tutorial

💡: Can LLMs utilize external tools to not only expand their capacities but also to make our NLP systems more robust, scalable, and interpretable?

🎯: Neuro-symbolic approaches in the era of large language models (LLMs): LLMs + tool use or tool augmented LLMs.

Paper collection on LLM + tool use

https://github.com/xlang-ai/llm-tool-use
Tutorial agenda

● Motivations
● Introduction to LLM + tool use
  ○ LLM + tool use in the perspective of executable language grounding
  ○ LLM + tool use examples
● Recent methods of LLM + tool use
  ○ LLM prompting for tool use
  ○ LLM finetuning/pretraining for tool use
● Other recent related work
  ○ Tool making
  ○ Planning
  ○ Code generation
● Challenges and future work
Human + tool use: motivations

- As humans, we have limited time and memory, feel tired, and have emotions.
- Human + tool use
  - Enhanced scalability
  - Improved consistency
  - Greater interpretability
  - Higher capacity and productivity

LLMs + tool use: motivations

- Just like humans, LLMs suffer from the similar limitations. But in the same way,

- LLMs + tool use
  - Enhanced scalability
  - Improved consistency
  - Greater interpretability
  - Higher capacity and productivity
Tutorial agenda

- Motivations
- Introduction to LLM + tool use
  - LLM + tool use in the perspective of executable language grounding
  - LLM + tool use examples
- Recent methods of LLM + tool use
  - LLM prompting for tool use
  - LLM finetuning/pretraining for tool use
- Other recent related work
  - Tool making
  - Planning
  - Code generation
- Challenges and future work
LLMs + tool use in perspective of executable language grounding

Ground language models into executable actions

- Mapping natural language instructions into code or actions executable within various environments such as databases, web applications, and robotic physical world.
- LM (planning and reasoning) + actions

Data analysis

Web/Apps

Robotic physical world

https://openai.com/blog/chatgpt-plugins
https://code-as-policies.github.io/
LLMs + tool use in perspective of executable language grounding

LLMs + tool use in executable language grounding tasks

**Inputs**

- **Language**: user question/request
- **Toolkit**: code, APIs to search engines, self-defined functions, expert models...
- **Environment**: databases, IDE, web/apps, visual and robotic physical world...

**Outputs**

- Grounded reasoning code/action seq that can be executed in the corresponding environment
  - What tools to select, when and how to use the selected tools
Example of LLMs + tool use in executable language grounding

LLMs + tool use in executable language grounding

- **Language**: user question/statement about a database
- **Toolkit**: code, APIs to NLP functionalities (expert models)
- **Environment**: databases, SQL/Python IDEs

Binder: Binding Language Models in Symbolic Languages

Zhoujun Cheng\(^1,2\), Tianbao Xie\(^4\), Peng Shi\(^5\), Chengzu Li\(^1\), Rahul Nadkarni\(^3\), Yushi Hu\(^3\), Caiming Xiong\(^6\), Dragomir Radev\(^7\), Mari Ostendorf\(^3\), Luke Zettlemoyer\(^3,8\), Noah A. Smith\(^3,4\), Tao Yu\(^1,3\)

\(^1\)The University of Hong Kong, \(^2\)Shanghai Jiao Tong University, \(^3\)University of Washington, \(^4\)Allen Institute for AI, \(^5\)University of Waterloo, \(^6\)Salesforce Research, \(^7\)Yale University, \(^8\)Meta AI

Project website: [https://lm-code-binder.github.io](https://lm-code-binder.github.io), ICLR 2023
Which is the best-selling shirt made in North America and with no chemicals?

<table>
<thead>
<tr>
<th>Shirt</th>
<th>Made_in</th>
<th>Sales</th>
<th>Price</th>
</tr>
</thead>
<tbody>
<tr>
<td>Polo</td>
<td>U.S.</td>
<td>500</td>
<td>$29.8</td>
</tr>
<tr>
<td>Luckyland</td>
<td>U.S.</td>
<td>900</td>
<td>$19.8</td>
</tr>
<tr>
<td>Timber Bay</td>
<td>Mexico</td>
<td>400</td>
<td>$25.9</td>
</tr>
<tr>
<td>Office</td>
<td>Turkey</td>
<td>600</td>
<td>$31.8</td>
</tr>
</tbody>
</table>

Details:
- 100% cotton
- 220GSM (6.5 oz)
- Natural stretch for comfortable fit

Details:
- 90% cotton, 10% polyester

Knowledge:

Question:

Input: Binding Language Models in Symbolic Languages
Which is the best-selling shirt made in North America and with no chemicals?

<table>
<thead>
<tr>
<th>Shirt</th>
<th>Made_in</th>
<th>Sales</th>
<th>Price</th>
</tr>
</thead>
<tbody>
<tr>
<td>Polo</td>
<td>U.S.</td>
<td>500</td>
<td>$29.8</td>
</tr>
<tr>
<td>Luckyland</td>
<td>U.S.</td>
<td>900</td>
<td>$19.8</td>
</tr>
<tr>
<td>Timber Bay</td>
<td>Mexico</td>
<td>400</td>
<td>$25.9</td>
</tr>
<tr>
<td>Office</td>
<td>Turkey</td>
<td>600</td>
<td>$31.8</td>
</tr>
</tbody>
</table>

Details:
- 100% cotton
- 220GSM (6.5 oz)
- Natural stretch for comfortable fit...

Pros:
- General

Cons:
- Interpretable
- Scalable
- Robust
Which is the best-selling shirt made in North America and with no chemicals?

<table>
<thead>
<tr>
<th>Shirt</th>
<th>Made_in</th>
<th>Sales</th>
<th>Price</th>
</tr>
</thead>
<tbody>
<tr>
<td>Polo</td>
<td>U.S.</td>
<td>500</td>
<td>$29.8</td>
</tr>
<tr>
<td>Luckyland</td>
<td>U.S.</td>
<td>900</td>
<td>$19.8</td>
</tr>
<tr>
<td>Timber Bay</td>
<td>Mexico</td>
<td>400</td>
<td>$25.9</td>
</tr>
<tr>
<td>Office</td>
<td>Turkey</td>
<td>600</td>
<td>$31.8</td>
</tr>
</tbody>
</table>

Details
- 100% cotton
- 220GSM (6.5 oz)
- Natural stretch for comfortable fit...

Details
- 90% cotton, 10% polyester
- ...

Knowledge:
Polo, Luckyland, Timber Bay are from North America. Polo and Office are of no chemicals. Luckyland has the most sales of 900. So the shirt is Luckyland.

Pros
- Improved but still...

Cons
- Interpretable
- Unreliable answer entailment
- Scalable
- Robust
### Which is the best-selling shirt made in North America and with no chemicals?

<table>
<thead>
<tr>
<th>Shirt</th>
<th>Made_in</th>
<th>Sales</th>
<th>Price</th>
</tr>
</thead>
<tbody>
<tr>
<td>Polo</td>
<td>U.S.</td>
<td>500</td>
<td>$29.8</td>
</tr>
<tr>
<td>Luckyland</td>
<td>U.S.</td>
<td>900</td>
<td>$19.8</td>
</tr>
<tr>
<td>Timber Bay</td>
<td>Mexico</td>
<td>400</td>
<td>$25.9</td>
</tr>
<tr>
<td>Office</td>
<td>Turkey</td>
<td>600</td>
<td>$31.8</td>
</tr>
</tbody>
</table>

**Details**
- **Polo**
  - 100% cotton
  - 220GSM (6.5 oz)
  - Natural stretch for comfortable fit...

- **Luckyland**
  - 90% cotton, 10% polyester
  - ...

- **Timber Bay**
  - Mexico
  - 400
  - $25.9

- **Office**
  - Turkey
  - 600
  - $31.8

---

**End-to-End**

**Input**

**Knowledge:**

- 100% cotton
- 220GSM (6.5 oz)
- Natural stretch for comfortable fit...

**Question:** Which is the best-selling shirt made in North America and with no chemicals?

**Reasoning path**

- Polo, Luckyland, Timber Bay are from North America.
- Polo and Office are of no chemicals.
- Luckyland has the most sales of 900. So the shirt is Luckyland.

---

**Chain-of-Thought**

**Input**

- GPT-3 Codex

**Reasoning path**

- Polo, Luckyland, Timber Bay are from North America.
- Polo and Office are of no chemicals.
- Luckyland has the most sales of 900. So the shirt is Luckyland.

---

**Semantic Parsing/Code Generation**

**Input**

- GPT-3 Codex

**SQL**

```sql
SELECT Shirt FROM T WHERE Shirt NOT LIKE 'Chemicals%' AND Made_in = 'North America' ORDER BY Sales DESC LIMIT 1
```

**Pros**

- Interpretable
- Scalable
- Robust

**Cons**

- Capable

---

**Binding Language Models in Symbolic Languages**

- **Interpretable**
- **Scalable**
- **Robust**

- **Uninterpretable & not robust**
- **Unreliable answer entailment**

- **Tool use:** SQL/Python
Which is the best-selling shirt made in North America and with no chemicals?

<table>
<thead>
<tr>
<th>Shirt</th>
<th>Made_in</th>
<th>Sales</th>
<th>Price</th>
</tr>
</thead>
<tbody>
<tr>
<td>Polo</td>
<td>U.S.</td>
<td>500</td>
<td>$29.8</td>
</tr>
<tr>
<td>Luckyland</td>
<td>U.S.</td>
<td>900</td>
<td>$19.8</td>
</tr>
<tr>
<td>Timber Bay</td>
<td>Mexico</td>
<td>400</td>
<td>$25.9</td>
</tr>
<tr>
<td>Office</td>
<td>Turkey</td>
<td>600</td>
<td>$31.8</td>
</tr>
</tbody>
</table>

Details:
- Polo: 100% cotton, 220GSM (6.5 oz), natural stretch for comfortable fit...
- Luckyland: 90% cotton, 10% polyester...
- Timber Bay:...
- Office:...

End-to-End Reasoning path:
Polo, Luckyland, Timber Bay are from North America. Polo and Office are of no chemicals. Luckyland has the most sales of 900. So the shirt is Luckyland.

Why LLM + tool use?

Pros:
- Interpretable
- Scalable
- Robust
- Capable

Tool use:
- SQL/Python + NLP expert model APIs
**PAL: Program-aided Language Models**

**PoT: Program of Thoughts Prompting: Disentangling Computation from Reasoning for Numerical Reasoning Tasks**
LLM + APIs to search/browser for gathering information

Retrieval augmented language models

- Covered in Yuchen and Michi’s sessions
- Another ACL tutorial by Akari Asai, Danqi Chen, Sewon Min, Zexuan Zhong

WebGPT: Browser-assisted question-answering with human feedback

ACL 2023 Tutorial:
Retrieval-based Language Models and Applications

ChatGPT + browsing
LLM + webs/apps or personalized functions

ChatGPT + Plugins: third-party apps/webs, Function calling
HuggingGPT: Solving AI Tasks with ChatGPT and its Friends in HuggingFace
Chameleon: Plug-and-Play Compositional Reasoning with Large Language Models
TaskMatrix.AI: Completing Tasks by Connecting Foundation Models with Millions of APIs
LLM + code, robotic arm, expert models: Code as Policies

```python
block_names = detect_objects("blocks")
bowl_names = detect_objects("bowls")
for bowl_name in bowl_names:
    if is_empty(bowl_name):
        empty_bowl = bowl_name
        break
objs_to_stack = [empty_bowl] + block_names
stack_objects(objs_to_stack)

def is_empty(name):
    pass

def stack_objects(obj_names):
    n_objs = len(obj_names)
    for i in range(n_objs - 1):
        obj0 = obj_names[i + 1]
        obj1 = obj_names[i]
        pick_place(obj0, obj1)
```
Tutorial agenda

● Motivations

● Introduction to LLM + tool use
  ○ LLM + tool use in the perspective of executable language grounding
  ○ LLM + tool use examples

● Recent methods of LLM + tool use
  ○ LLM prompting for tool use
  ○ LLM finetuning/pretraining for tool use

● Other recent related work
  ○ Tool making
  ○ Planning
  ○ Code generation

● Challenges and future work
## LLM prompting for tool use

**Chain-of-Thought (Wei et al., 2022)**

Q: Roger has 5 tennis balls. He buys 2 more cans of tennis balls. Each can has 3 tennis balls. How many tennis balls does he have now?

A: Roger started with 5 tennis balls. 2 cans of 3 tennis balls each is 6 tennis balls. $5 + 6 = 11$. The answer is 11.

Q: The bakers at the Beverly Hills Bakery baked 200 loaves of bread on Monday morning. They sold 93 loaves in the morning and 39 loaves in the afternoon. A grocery store returned 6 unsold loaves. How many loaves of bread did they have left?

A: The bakers started with 200 loaves. They sold 93 in the morning and 39 in the afternoon. So they sold $93 + 39 = 132$ loaves. The grocery store returned 6 unsold loaves. So they had $200 - 132 = 68$ loaves left. The answer is 68.

**Program-aided Language models (this work)**

Q: Roger has 5 tennis balls. He buys 2 more cans of tennis balls. Each can has 3 tennis balls. How many tennis balls does he have now?

A: Roger started with 5 tennis balls. 2 cans of 3 tennis balls each is 6 tennis balls. tennis_balls = 5 2 cans of 3 tennis balls each is bought_balls = 2 * 3 tennis_balls. The answer is answer = tennis_balls + bought_balls

Q: The bakers at the Beverly Hills Bakery baked 200 loaves of bread on Monday morning. They sold 93 loaves in the morning and 39 loaves in the afternoon. A grocery store returned 6 unsold loaves. How many loaves of bread did they have left?

A: The bakers started with 200 loaves loaves_baked = 200 They sold 93 in the morning and 39 in the afternoon loaves_sold_morning = 93 loaves_sold_afternoon = 39 The grocery store returned 6 loaves loaves Returned = 6 The answer is answer = loaves_baked - loaves_sold_morning - loaves_sold_afternoon + loaves Returned

**PAL: Program-aided Language Models**
Which is the best-selling shirt made in North America and with no chemicals?

<table>
<thead>
<tr>
<th>Shirt</th>
<th>Made_in</th>
<th>Sales</th>
<th>Price</th>
</tr>
</thead>
<tbody>
<tr>
<td>Polo</td>
<td>U.S.</td>
<td>500</td>
<td>$29.8</td>
</tr>
<tr>
<td>Luckyland</td>
<td>U.S.</td>
<td>900</td>
<td>$19.8</td>
</tr>
<tr>
<td>Timber Bay</td>
<td>Mexico</td>
<td>400</td>
<td>$25.9</td>
</tr>
<tr>
<td>Office</td>
<td>Turkey</td>
<td>600</td>
<td>$31.8</td>
</tr>
</tbody>
</table>

Details:
- 100% cotton
- 220GSM (6.5 oz)
- Natural stretch for comfortable fit

Details:
- 90% cotton, 10% polyester

---

# WikITQ
Generate SQL given the question and table to answer the question correctly.

```sql
CREATE TABLE Electoral district of Lachlan(
  row_id int,
  member text,
  party text,
  term text)
/
3 example rows:
SELECT * FROM w LIMIT 3;
row_id member party term
0 john ryan none 1859-1864
1 james martin none 1864-1869
2 james watson none 1869-1880
/
Q: of the members of the third incarnation of the lachlan, who served the longest?
Binder: SELECT member FROM w ORDER BY f("How long does it last?", term) DESC LIMIT 1
```
ReAct: Synergizing Reasoning and Acting in Language Models

**LLM + tool use for QA and decision making: ReAct**

---

**Act 1:** Think [First I need to find a pepper shaker... more likely to appear in cabinets (1-6), countertops (1-3), ...]

**Act 2:** Go to cabinet 1

**Act 3:** Go to countertop 3

**Act 4:** Take pepper shaker 1 from sinkbasin 1

**Act 5:** Take pepper shaker 1 from sinkbasin 1

---

**Act 1:** Go to drawer 1

**Act 2:** Open drawer 1

**Act 3:** Go to sinkbasin 1

**Act 4:** Take peppershaker 1 from sinkbasin 1

**Act 5:** Nothing happens.

**Act 6:** On the cabinet 1, you see a vase 2.

**Act 7:** Pick up the pepper shaker 1 from the countertop 3.

**Act 8:** Think [Now I find a pepper shaker 1. Next, I need to put it in/on drawer 1.]

---

**Act 9:** Go to drawer 1

**Act 10:** Open drawer 1

**Act 11:** Put pepper shaker 1 in/on drawer 1

**Act 12:** You put pepper shaker 1 in/on the drawer 1.
LLM finetuning/pretraining for tool use: TALM

TALM: Tool Augmented Language Models

Toolformer: Language Models Can Teach Themselves to Use Tools
MRKL Systems: A modular, neuro-symbolic architecture that combines large language models, external knowledge sources and discrete reasoning
LLM finetuning/pretraining for tool use: 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.
LLM finetuning/pretraining for tool use: Toolformer

1. Sample API Calls
   - \( x_{i-1} \) = Pittsburgh is also known as
   - \( c_i^1 \) = What other name is Pittsburgh known by?
   - \( c_i^2 \) = Which country is Pittsburgh in?
   - \( r_i^1 \) = Steel City
   - \( r_i^2 \) = United States

2. Execute API Calls
   - \( L_i(c_i^1 \rightarrow \text{Steel City}) \)
   - \( L_i(c_i^1 \rightarrow \varepsilon) \leq L_i(\varepsilon) \)
   - \( L_i(c_i^2 \rightarrow \text{United States}) \)
   - \( L_i(c_i^2 \rightarrow \varepsilon) \geq L_i(\varepsilon) \)

3. Filter API Calls

LM Dataset with API Calls

\( x^* \) = Pittsburgh is also known as [QA(What ...? \rightarrow Steel City)] the Steel City.
Tutorial agenda

- Motivations
- Introduction to LLM + tool use
  - LLM + tool use in the perspective of executable language grounding
  - LLM + tool use examples
- Recent methods of LLM + tool use
  - LLM prompting for tool use
  - LLM finetuning/pretraining for tool use
- Other recent related work
  - Tool making
  - Planning
  - Code generation
- Challenges and future work
Other recent related work: LLM as tool maker

LATM: Large Language Models as Tool Makers
Other recent related work: planning via classical resolver

LLM+P: Empowering Large Language Models with Optimal Planning Proficiency
PDDL Planning with Pretrained Large Language Models
Other recent related work: code generation

- CodeT: Code Generation with Generated Tests
- Teaching Large Language Models to Self-Debug
- DocPrompting: Generating Code by Retrieving the Docs
- Coder Reviewer Reranking for Code Generation
Challenges and future work

- **Complexity**: more complex domain professional/unseen tools?
- **Interactivity**: go beyond single turn?
- **Evaluation**: multiple possible solutions? Real-time interactive evaluation?
- **Efficiency**: smaller models?
- **Reliability**: know when to abstain, know its capacity, memorizing and querying tools?
- **Others**
  - Better tool API design/tool making?
  - Personalization?
  - ......
Thank you!

General tutorial site
https://wenting-zhao.github.io/complex-reasoning-tutorial/

Paper collection on LLM + tool use
https://github.com/xlang-ai/llm-tool-use
References

[2] Do As I Can, Not As I Say: Grounding Language in Robotic Affordances
[5] Inner Monologue: Embodied Reasoning through Planning with Language Models
[10] Synergizing Reasoning and Acting in Language Models
[14] Program of Thoughts Prompting: Disentangling Computation from Reasoning for Numerical Reasoning Tasks
[15] Planning with Large Language Models via Corrective Re-prompting
References

[17] LLM-Planner: Few-Shot Grounded Planning for Embodied Agents with Large Language Models
[18] Don't Generate, Discriminate: A Proposal for Grounding Language Models to Real-World Environments
[19] Large language models are versatile decomposers: Decompose evidence and questions for table-based reasoning
[20] Toolformer: Language Models Can Teach Themselves to Use Tools
[22] Grounding Large Language Models in Interactive Environments with Online Reinforcement Learning
[23] Grounded Decoding: Guiding Text Generation with Grounded Models for Robot Control
[27] HuggingGPT: Solving AI Tasks with ChatGPT and its Friends in HuggingFace
[28] TaskMatrix.AI: Completing Tasks by Connecting Foundation Models with Millions of APIs
[29] ART: Automatic multi-step reasoning and tool-use for large language models
[30] API-Bank: A Benchmark for Tool-Augmented LLMs
Tool Learning with Foundation Models
Chameleon: Plug-and-Play Compositional Reasoning with Large Language Models
GeneGPT: Augmenting Large Language Models with Domain Tools for Improved Access to Biomedical Information
LLM as A Robotic Brain: Unifying Egocentric Memory and Control
Voyager: An Open-Ended Embodied Agent with Large Language Models
GPT4Tools: Teaching Large Language Model to Use Tools via Self-instruction
PEARL: Prompting Large Language Models to Plan and Execute Actions Over Long Documents
Large Language Models as Tool Makers
Gorilla: Large Language Model Connected with Massive APIs
On the Tool Manipulation Capability of Open-source Large Language Models
Making Language Models Better Tool Learners with Execution Feedback
Small models are valuable plug-ins for large language models
Hierarchical Prompting Assists Large Language Model on Web Navigation
Multimodal Web Navigation with Instruction-Finetuned Foundation Models
ToolkenGPT: Augmenting Frozen Language Models with Massive Tools via Tool Embeddings
CRITIC: Large Language Models Can Self-Correct with Tool-Interactive Critiquing
SheetCopilot: Bringing Software Productivity to the Next Level through Large Language Models
References

[49] SQL-PaLM: Improved Large Language Model Adaptation for Text-to-SQL
[50] From Pixels to UI Actions: Learning to Follow Instructions via Graphical User Interfaces
[51] Modular Visual Question Answering via Code Generation
[52] ToolAlpaca: Generalized Tool Learning for Language Models with 3000 Simulated Cases
[53] Mind2Web: Towards a Generalist Agent for the Web
[54] LLM+P: Empowering Large Language Models with Optimal Planning Proficiency
[56] OpenAGI: When LLM Meets Domain Experts
[57] Can LLM Already Serve as A Database Interface? A Big Bench for Large-Scale Database Grounded Text-to-SQLs
[58] Leveraging Pre-trained Large Language Models to Construct and Utilize World Models for Model-based Task Planning
[59] ReWOO: Decoupling Reasoning from Observations for Efficient Augmented Language Models
[60] ToolCoder: Teach Code Generation Models to use API search tools
[61] Chameleon: Plug-and-Play Compositional Reasoning with Large Language Models
Other proof-of-concept projects

- Auto-GPT
- LangChain Agent
- ChatGPT plugins
- BabyAGI
- GPT-Engineer
- ToolBench, BMTools
- ...
