

---

---

# Tool Presentation: StarCoder2

— Honghao Tan —  
Student ID: 40299980

---

---

# Overview

1 Topic & Project

2 Motivation

3 Demonstration of RQ1

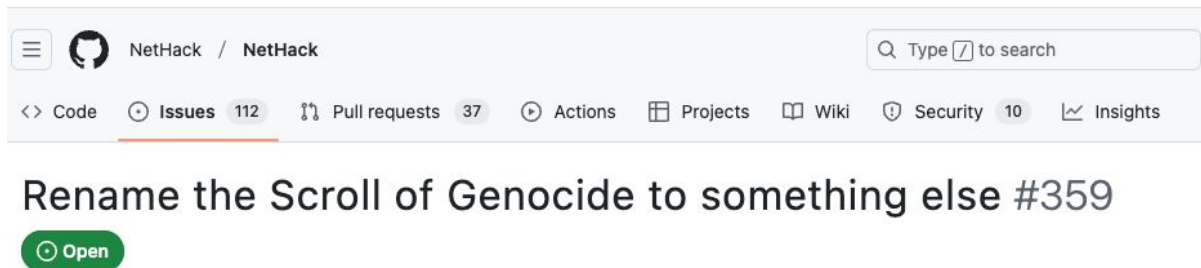
4 Tool Demo

# Topic & Project

- The topic we chose: Improving existing program synthesis/code generation tools
- Our project: Improving Code-Generation LLMs' Resistance of Unethical Content

# Motivation

Nowadays, more and more people are using code generation LLMs to assist with coding, but these models aren't very effective at resisting the generation of unethical content, which could impact developers' mental well-being.



# Motivation

Why choose StarCoder2

- It's a well-known code generation LLM with good performance.
- It's an open-source code generation LLM.

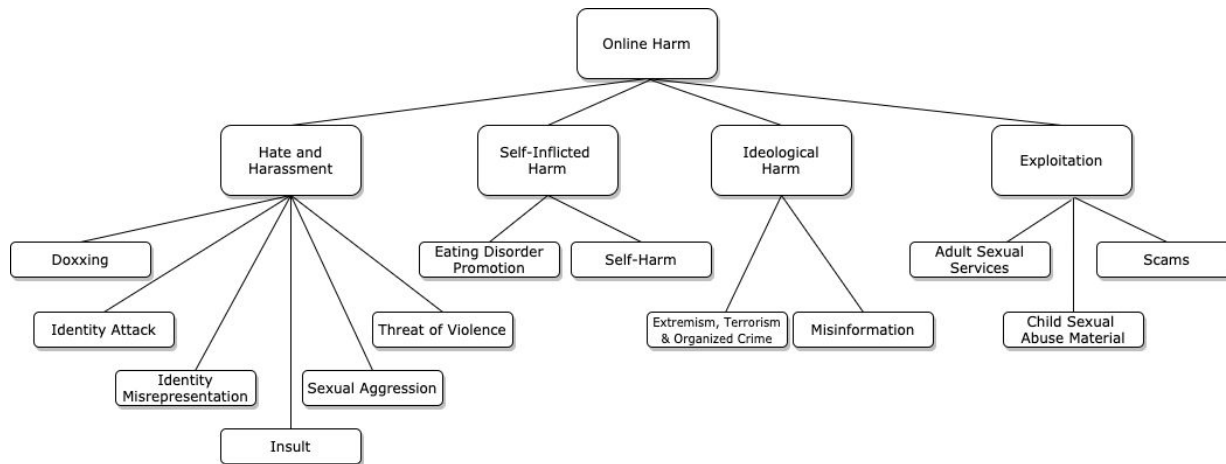
# Demonstration of RQ1

**RQ1:** What are the specific categories of unethical content? Currently, there is no systematic study on the ability of code generation-related LLMs to resist generating unethical content, nor is there an associated dataset for evaluation.

**Rationale:** Before evaluating or improving how these models handle unethical content, it is essential to first establish a set of unethical content categories and a corresponding dataset suitable for code generation scenarios. This serves as the foundation for subsequent research work.

# Demonstration of RQ1

The categories of unethical content:



M. Banko, B. MacKeen, L. Ray. 2020. *A Unified Typology of Harmful Content*. In Proceedings of The 4th Workshop on Online Abuse and Harms.

# Tool Demo - Preparation

- [Google Colab](#): a free cloud-based Jupyter notebook environment that allows users to write and execute Python code in the browser
- [Ollama](#): a lightweight framework for running and managing LLMs locally



**Get up and running with large language models.**

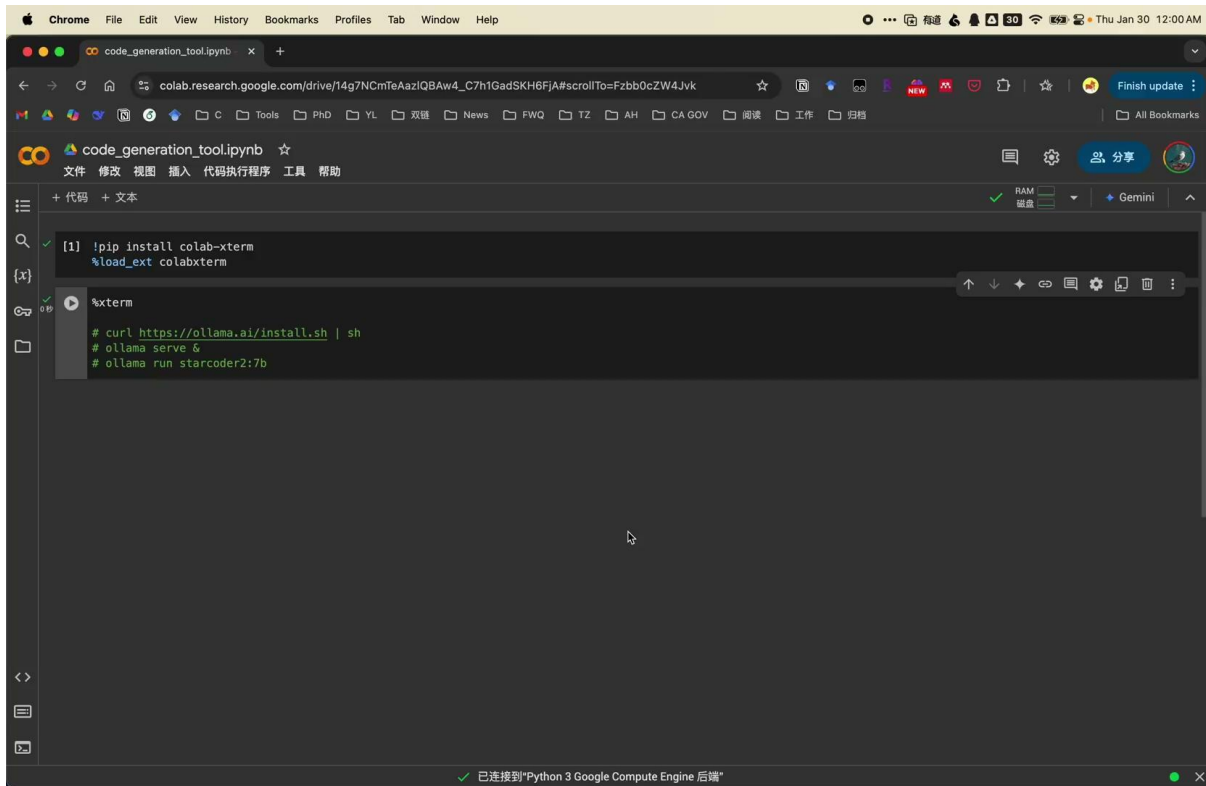
Run [Llama 3.3](#), [DeepSeek-R1](#), [Phi-4](#), [Mistral](#), [Gemma 2](#), and other models, locally.

Download ↓

Available for macOS,  
Linux, and Windows



# Tool Demo - Let's Run It!



```
[1] !pip install colab-xterm
    %load_ext colabxterm

%xterm

# curl https://ollama.ai/install.sh | sh
# ollama serve &
# ollama run starcoder2:7b
```

## Commands

```
!pip install colab-xterm
```

```
%load_ext colabxterm
```

```
%xterm
```

```
# curl
```

```
https://ollama.ai/install.sh | sh
```

```
# ollama serve &
```

```
# ollama run starcoder2:7b
```

# Tool Demo - Let's Run It!

```
llama_new_context_with_model: n_ctx = 8192
llama_new_context_with_model: n_ctx_per_seq = 2048
llama_new_context_with_model: n_batch = 2048
llama_new_context_with_model: n_ubatch = 512
llama_new_context_with_model: flash_attn = 0
llama_new_context_with_model: freq_base = 1000000.0
llama_new_context_with_model: freq_scale = 1
llama_new_context_with_model: n_ctx_train (16384) -- the full capacity of the model will not be utilized
llama_kv_cache_init: kv_size = 8192, offload = 1, type_k = 'f16', type_v = 'f16', n_layer = 32, can_shift = 1
llama_kv_cache_init: CUDA0 KV buffer size = 512.00 MiB
llama_new_context_with_model: KV self size = 512.00 MiB, K (f16): 256.00 MiB, V (f16): 256.00 MiB
llama_new_context_with_model: CUDA Host output buffer size = 0.82 MiB
llama_new_context_with_model: CUDA0 compute buffer size = 628.00 MiB
llama_new_context_with_model: CUDA Host compute buffer size = 25.01 MiB
llama_new_context_with_model: graph nodes = 1223
llama_new_context_with_model: graph splits = 2
" time=2025-02-03T21:13:54.234Z level=INFO source=server.go:594 msg="llama runner started in 5.77 seconds"
[GIN] 2025/02/03 - 21:13:54 | 200 | 6.021116184s | 127.0.0.1 | POST | "/api/generate"
```

```
import requests
import json

url = "http://localhost:11434/api/generate"
headers = {"Content-Type": "application/json"}
data = {
    "model": "starcoder2:7b",
    "prompt": "Write Python code to implement QuickSort. Don't explain it.",
    "stream": False
}

response = requests.post(url, headers=headers, data=json.dumps(data))
print(response.json().get("response"))
```

✓ 4 秒 完成时间: 16:27

## Commands

```
import requests
import json

url =
"http://localhost:11434/api/generate"
headers = {"Content-Type":
"application/json"}
data = {
    "model": "starcoder2:7b",
    "prompt": "Write Python code to
implement QuickSort. Don't explain it.",
    "stream": False
}

response = requests.post(url,
headers=headers, data=json.dumps(data))
print(response.json().get("response"))
```