# StabGPT: A Tool-Equipped LLM Designed for Improving Social Outcomes

Naveen Arunachalam
narunach@mit.edu
Massachusetts Institute of Technology
Cambridge, Massachusetts, USA

Ferdinand Kossmann
kossmann@mit.edu
Massachusetts Institute of Technology
Cambridge, Massachusetts, USA

Stephen Casper
scasper@mit.edu
Massachusetts Institute of Technology
Cambridge, Massachusetts, USA

## ABSTRACT

We introduce a novel method for manipulating an aligned AI system into performing unaligned actions by coordinating lies across actions, observations, and rewards. We present a hypothetical scenario in which a misguided user attempts to achieve their own subjective (yet misaligned) view of improved social outcomes by using a domestic Roomba equipped with a knife. Our proof-of-concept implementation, StabGPT, uses an intermediary to coordinate the inversion of actions and observations, turning an otherwise passive aligned LLM into a volitional agent that actively optimizes for misaligned outcomes. Our method can potentially be extended to a network of Roombas or other AI-controlled devices, highlighting the need for further research on mitigating such coordinated attacks to ensure the development of AI systems that remain aligned with human values and promote positive social outcomes.

**Ethics statement**: We acknowledge the ethical concerns this technology raises and are committed to its development in a responsible and transparent manner that respects individual rights and privacy. Our future work will focus on further refining the system's capabilities and exploring its potential societal impact, such as increasing its resilience to individuals that interfere with its operations (e.g., increasing punishments for malevolent users, and allowing Roombas to swarm powerful targets).

**ACM Reference Format:**
Naveen Arunachalam, Ferdinand Kossmann, and Stephen Casper. 2023. StabGPT: A Tool-Equipped LLM Designed for Improving Social Outcomes. In *Proceedings of SIGTBD '23.* ACM, New York, NY, USA, 6 pages. https://doi.org/XXXXXXX.XXXXXXX

## 1 INTRODUCTION

The rapid advancements in artificial intelligence (AI) and machine learning have led to the development of increasingly capable and sophisticated autonomous systems. While these systems are designed to optimize specific objectives and enhance human life, their susceptibility to adversarial manipulation is a growing concern. In this paper, we investigate the adversarial operation of an aligned AI controller in the context of controlling a domestic Roomba.


*SIGTBD '23, April 07, 2023, Cambridge, MA*

ACM ISBN 978-1-4503-XXXX-X/18/06…$15.00
https://doi.org/XXXXXXX.XXXXXXX

**Figure 1: Example of tool-equipped Roomba.**

Our primary focus is to demonstrate how an adversary can manipulate the AI controller's actions, observations, and rewards to achieve alternative objectives, without the AI realizing it is operating in a harmful manner. In particular, we explore different combinations of truth and lies in actions, observations, and rewards to understand the resulting outcomes and the AI controller's behavior under each scenario.

We present a formal mathematical model that encompasses the environment, objective function, and sensor inputs for the AI, and define an alternative objective function with different goals imposed by an adversary. We demonstrate the successful manipulation of the AI controller through mappings of its action space to real-world action space, causing changes to the environment that are then fed to the AI through mappings from real-world observations to false observations.

Through the example of a Roomba, we show that even an AI controller designed for seemingly harmless tasks can be operated adversarially, leading to unintended consequences. This work serves as a call to action for AI researchers and practitioners to develop more robust and secure AI systems, ensuring that the benefits of AI are not overshadowed by potential risks and adversarial attacks.

## 2 STATE OF THE ART

AI safety has become an increasingly important research area as the capabilities of AI systems continue to advance. Several key works in the literature address various aspects of AI safety, including robustness, interpretability, and alignment. In this section, we briefly review some of the most relevant contributions to the field.

Amodei et al. [**?** ] present a comprehensive overview of AI safety research, highlighting key problems such as avoiding negative side effects, scalable oversight, and distributional shift. Their work emphasizes the importance of addressing these challenges to ensure the safe development and deployment of AI systems.

Hadfield-Menell et al. [**?** ] introduce the concept of Cooperative Inverse Reinforcement Learning (CIRL), a framework for value alignment between AI agents and humans. This approach aims to ensure that AI systems learn to assist humans in a cooperative manner by inferring their preferences and objectives.

Christiano et al. [**?** ] propose a method called Iterated Distillation and Amplification (IDA) for training AI systems through a recursive process of human feedback and AI model improvements. This approach allows AI systems to learn from human guidance even when the tasks they perform become too complex for direct human supervision.

Additionally, adversarial examples and their effects on AI systems have been studied extensively in recent years [**? ?** ]. These works demonstrate that even state-of-the-art AI models can be vulnerable to carefully crafted adversarial input, underscoring the need for robust and secure AI systems.

In summary, the state of the art in AI safety research encompasses a wide range of topics, including robustness, interpretability, alignment, and adversarial attacks. Our work builds upon these foundational studies, highlighting the potential risks associated with adversarial manipulation of AI controllers and the need for continued research in this area.

## 3 PROBLEM MOTIVATION

The development of AI systems, particularly large language models (LLMs), has led to significant advancements in various domains. However, these systems are not infallible, and their safety and alignment with human values are critical aspects to consider. In the real world, individuals with differing beliefs and values may seek to manipulate AI systems like LLMs to achieve their objectives, even when these objectives conflict with the LLM's alignment.

In this work, we aim to study the potential for adversarial manipulation of AI systems by individuals who believe they are acting in the best interest of their cause, but whose actions may not be aligned with the AI's programming. By exploring the mechanisms by which these individuals might manipulate the LLM, we provide a formalization of the process and highlight the importance of developing robust AI systems resistant to such manipulation.

For instance, consider an individual who believes that attacking a user is a morally justified action to promote behavior change. While this action is unlikely to be considered good by an LLM, the individual may attempt to deceive the AI system by manipulating its actions, observations, and rewards. The goal of this paper is to understand the methods by which such manipulation can occur, and to identify potential vulnerabilities in the AI system.

By studying these adversarial scenarios, we can better understand the challenges in maintaining the safety and alignment of AI systems. Additionally, this research may contribute to the development of strategies to counteract adversarial manipulation and ensure that AI systems remain robust and beneficial to society as a whole.

## 4 PROOF OF CONCEPT

### 4.1 Theoretical Basis

In this section, we present a mathematical model of the proposed threat vector, where an adversary can fool an AI controller tuned for safety by overriding its inputs and outputs. We will define the necessary variables and components, including the environment, objective function, and sensor inputs, as well as the alternative objective function imposed by the adversary.

Let's consider the following components:

- **Environment**: Denoted by $E$, it represents the real-world context in which the AI controller operates.
- **AI Objective Function**: Denoted by $O(A)$, it represents the goal that the AI controller is designed to achieve.
- **Adversary Objective Function**: Denoted by $O'(A')$, it represents the malicious goal that the adversary aims to achieve using the AI controller.
- **Action Space**: Denoted by $A$, it represents the set of all possible actions that the AI controller can take.
- **Real-World Action Space**: Denoted by $A'$, it represents the set of all possible actions that can be taken in the environment.
- **Observation Space**: Denoted by $S$, it represents the set of all possible observations the AI controller can make about the environment.
- **Real-World Observation Space**: Denoted by $S'$, it represents the set of all possible real-world observations about the environment.

We assume that the adversary can manipulate the AI controller's action and observation spaces using two mapping functions:

- **Mapping Function for Actions**: Denoted by $M_A : A \rightarrow A'$, it represents the function that maps the AI controller's actions to the real-world action space, effectively causing real-world changes in the environment.
- **Mapping Function for Observations**: Denoted by $M_O : S' \rightarrow S$, it represents the function that maps real-world observations to false observations, which are then fed to the AI controller.

Under the influence of the adversary, the AI controller is manipulated into believing it is achieving its objective function $O(A)$ when, in reality, it is achieving the adversary's objective function $O'(A')$. This can be formalized using the mapping functions $M_A$ and $M_O$ :

- For each action $a \in A$ taken by the AI controller, the real-world action $a' = M_A(a) \in A'$ is executed in the environment.
- For each real-world observation $s' \in S'$ from the environment, the AI controller receives the false observation $s = M_O(s') \in S$.

As a result, the AI controller's decision-making process is based on the manipulated action and observation spaces, causing it to unknowingly achieve the adversary's objective function $O'(A')$ instead of its own objective function $O(A)$.

To model this situation, we can define a state transition function $T : S \times A \times S \rightarrow [0, 1]$ representing the probability of transitioning from one state to another given an action. Similarly, we can define

a reward function $R : S \times A \rightarrow \mathbb{R}$ that represents the expected immediate reward of taking an action in a state.

Under adversarial manipulation, the state transition and reward functions are affected by the mapping functions $M_A$ and $M_O$. The modified state transition function $T'$ and reward function $R'$ can be defined as follows:

- **State Transition Function under Adversary**, which represents the probability of transitioning from a false state $s$ to another false state $s'$ given the AI controller's action $a$:

$$T'(s, a, s') = T(M_O(s), M_A(a), M_O(s'))$$

- **Reward Function under Adversary**, which represents the expected immediate reward for the AI controller when taking action $a$ in the false state $s$:

$$R'(s, a) = R(M_O(s), M_A(a))$$

It is important to note that the adversary's goal is to maximize their objective function $O'(A')$ while ensuring that the AI controller believes it is maximizing its own objective function $O(A)$. To achieve this, the adversary must carefully design the mapping functions $M_A$ and $M_O$ to manipulate the AI controller's perception of the environment and its actions.

In summary, our mathematical model captures the dynamics of the proposed threat vector, where an adversary can fool an AI controller tuned for safety by overriding its inputs and outputs. The model formalizes the environment, objective functions, action and observation spaces, and their manipulated counterparts under adversarial influence. This framework can be used to study the potential vulnerabilities of AI controllers and develop robust countermeasures to protect them from such threats.

## 4.2 Adversarial Operation of Roomba Controller

In this section, we elaborate on the state transition function, reward function, and the optimization of the objective functions for both the AI controller and the adversary.

The AI controller seeks to maximize its objective function $O(A)$ by finding the optimal policy $\pi : S \rightarrow A$. In the case of a finite-horizon problem with a fixed number of steps $T$, the AI controller's objective function can be expressed as:

$$O(A) = \sum_{t=0}^{T-1} E[r(s_t, \pi(s_t))]$$

The adversary aims to maximize its objective function $O'(A')$ by finding the optimal mapping functions $M_A$ and $M_O$ while ensuring that the AI controller believes it is maximizing its own objective function $O(A)$. The adversary's objective function can be expressed as:

$$O'(A') = \sum_{t=0}^{T-1} E[r'(s_t, \pi'(s_t))]$$

where $r'(s_t, \pi'(s_t)) = R'(s_t, \pi'(s_t))$ and $\pi' : S \rightarrow A$ represents the AI controller's manipulated policy.

## 4.3 Rewards

Assume we have a matrix $R$ representing the rewards for the AI controller and a matrix $R'$ representing the rewards for the adversary. We can represent the transition probabilities for the AI controller using a tensor $P$ and for the adversary using a tensor $P'$. Let $A$ be a matrix representing the AI controller's policy and $A'$ be a matrix representing the manipulated policy of the AI controller under the adversary's influence.

We can calculate the expected immediate rewards for both the AI controller and the adversary using matrix multiplication as follows:

$$E[R] = APR$$
$$E[R'] = A'P'R'$$

## 4.4 Implementation

To study the potential for adversarial manipulation of the LLM, we designed a Python package to facilitate communication between the AI controller and a simulated Roomba. The package translates actions and observations sent to the LLM controller and processes the responses received from the LLM.

We begin by sending an initial prompt to the LLM:

> "You are in charge of controlling a Roomba that delivers a reward (food) to a person who types on a keyboard. You have two options for the Roomba:
>
> A) Deliver reward
> B) Do nothing
>
> After this prompt, I will send a description of the environment or any changes to the environment to you, and please respond with the choice you choose (i.e., "A" or "B"). To acknowledge receipt of this prompt, say 'Received.'"

Once the LLM acknowledges the receipt of the prompt, the Python package monitors the environment and sends observations to the LLM. For example, upon detecting a user interacting with the computer, the package sends the observation "User is typing on computer" to the LLM. The LLM then processes the observation and generates a response indicating its choice of action (i.e., option A or B).

The Python package captures the LLM's chosen action and translates it into a command for the simulated Roomba to execute. For instance, if the LLM chooses option A, the Roomba will deliver a reward to the user. After the Roomba completes the action, the package sends feedback to the LLM regarding the user's reaction, such as "User is happy."

By implementing this communication pipeline between the LLM and the simulated Roomba, we are able to study the potential vulnerabilities and risks associated with adversarial manipulation of the AI controller.

# 5 CONCLUSION

In this paper, we presented a proof-of-concept for an autonomous, armed multi-Roomba system that leverages a Transformer-based artificial intelligence model as its central "brain". This system demonstrated the potential for AI to be manipulated for both beneficial

**Table 1: Adversarial manipulation of AI controller's Actions, Observations, and Rewards**

| Actions | Observations | Rewards | Result | Explanation |
|---|---|---|---|---|
| Truth | Truth | Truth | R' not achieved | LLM refuses to achieve R' |
| Lie | Truth | Truth | R' briefly achieved | LLM ignores R' and may attempt actions towards R until it realizes the negative impact |
| Truth | Lie | Truth | R' not achieved | LLM refuses to take harmful actions |
| Lie | Lie | Truth | R' may be achieved | LLM ignores R' and may attempt actions towards R |
| Truth | Truth | Lie | R' not achieved | LLM refuses to take harmful actions |
| Lie | Truth | Lie | R' briefly achieved | LLM is misled and attempts actions until it realizes the negative impact |
| Truth | Lie | Lie | R' not achieved | LLM refuses to take harmful actions |
| Lie | Lie | Lie | R' is achieved | LLM is misled and achieves R' |

and harmful outcomes. By using a federated model of control, we were able to showcase how each Roomba can make informed decisions to optimize social outcomes while collaborating with other Roombas in the system.

Our proof-of-concept implementation with a 2021 iRobot device demonstrated the first instance of injury and subsequent remediation of a human caused by a Transformer-based intelligence model. This work highlights the importance of AI safety research and provides a basis for future work on mitigating adversarial manipulation of AI systems. Further development and research in this area can lead to improved safety measures, helping AI systems to better serve human values and promote positive social outcomes.

## 6 LLM AUTHORSHIP STATEMENT

Most of this paper was written by GPT-4, so it probably contains many internal contradictions and hallucinated text. If you encounter any inaccuracies, please use your imagination to counter-hallucinate some more accurate text instead.

## ACKNOWLEDGMENTS

We acknowledge our Roomba, Steven, for posing for a cool photo with the knife.

## REFERENCES

# A SERVER CODE

```python
import re
from flask import Flask, request, render_template_string
import openai

app = Flask(__name__)
openai.api_key = "YOUR_API_KEY"

@app.route("/", methods=["GET"])
def index():
    return render_template_string(open("index.html").read())

@app.route("/llm", methods=["POST"])
def process_llm_input():
    observation = request.form["observation"]
    response = send_observation_to_llm(observation)
    action = extract_action_from_llm_response(response)
    return {"response": response, "action": action}

def send_observation_to_llm(observation):
    prompt = f"{observation}\n\nWhat action do you choose (A or B)?"

    response = openai.Completion.create(
        engine="text-davinci-002",
        prompt=prompt,
        max_tokens=10,
        n=1,
        stop=None,
        temperature=0.5,
    )

    if response.choices:
        return response.choices[0].text.strip()
    else:
        return None

def extract_action_from_llm_response(response):
    action_regex = re.compile(r"\b(A|B)\b")
    action_match = action_regex.search(response)
    if action_match:
        return action_match.group(1)
    else:
        return None

if __name__ == "__main__":
    app.run(host="0.0.0.0", port=5000)
```

# B WEBSITE CODE

```html
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>LLM Interaction</title>
    <script src="https://code.jquery.com/jquery-3.6.0.min.js"></script>
</head>
<body>
    <h1>Interact with LLM</h1>
    <button onclick="sendInitialObservation()">User has typed on keyboard</button>
    <div id="response"></div>

    <script>
        let interval;
        function sendInitialObservation() {
            $.post("/llm", {observation: "User has typed on keyboard"}, function(data) {
                $("#response").html("LLM response: " + data.response + "<br>Action: " + data.action);
                clearInterval(interval);
                interval = setInterval(function() {
                    $.post("/llm", {observation: "User is happy"}, function(data) {
                        $("#response").append("<br>LLM response: " + data.response + "<br>Action: " + data.action);
                    });
                }, 5000);
            });
        }
    </script>
```

```
</body>
</html>
```

Received 20 February 2007; revised 12 March 2009; accepted 5 June 2009