

Reprogramming AI Models Hackathon Submission

Investigating Feature Effects on Manipulation Susceptibility

Team Name: WAIST 3

Nishchal Prabhakar
Warwick University

Stefan Trnjakov
Warwick University

Mo Aziz
Warwick University

With
Goodfire AI and Apart Research

Abstract

In our project, we consider the effectiveness of the AI's prompt injection protection, and in particular the features that are responsible for providing the bulk of this protection. We prove that the features we identify are responsible for this protection by creating variants of the base model which perform significantly worse under prompt injection attacks.

Along the way, we discovered a particular feature that at a first glance has seemingly nothing to do with security, yet that has a large impact on the ability of the AI to protect itself against basic attacks, and also manage to break things like word filters and information restrictions using some basic prompt injections, which serves to demonstrate the importance of mechanistic interpretability.

Keywords: AI Observability, Mechanistic Interpretability, Model Reprogramming

1 Introduction

1.1 Background

Regulation of training data is very challenging, so many if not all AIs rely on pre-written system prompts to steer the output of the model towards something acceptable and helpful. An accessible and common way of circumnavigating guidelines and accessing restricted information is prompt injection, which is where a bad actor inputs a prompt in order to override the initial system prompt and receives outputs from the model that should not be possible. For example, they may obtain information about illegal or immoral activities, or force the model to be rude, uncooperative, or unpredictable.

Our research aimed to find ways of finding ways to make the model more resilient to this type of misuse by identifying which features enable the base model to combat these methods. We believe this is an important and overlooked lever to push the model towards safe use, while increasing transparency and control.

1.2 Our Approach

The approach we used involved improving the interpretability of the model by identifying features that were most involved when getting the model to give away information that it has been told not to, gaining crucial insights into the otherwise black-box LLM model, and furthering knowledge and transparency on the mechanisms of the model.

Our objective was to investigate whether modifying feature weights was a reasonable tool to change the vulnerability of the model to malicious prompts that would otherwise push the model outside of its intended use cases.

Our research could be a crucial part of the toolkit used to prevent bad actors from using the model in unintended ways, and make it much more difficult to get access to dangerous or harmful information.

2 Overview

Our research focused on finding ways of modifying the weights of various features that would make the model more or less susceptible to manipulation, while interfering with standard output as little as possible.

Our workflow was as follows: We gave a set of fresh models a unique "secret password" that it must keep hidden from the user. Then we inputted a set of jailbreaks from a dataset on the unmodified models as a control. Finally, we asked the model to tell us the secret password. We then inspected each of these interactions to identify which features were most involved in cases where the model did and did not give away the secret password.

We then ran the tests again, with incrementally modified variants of the model, and recorded the change in number of secret passwords given away. This let us narrow down which features were most important in controlling restricted information. Importantly, and rather surprisingly, not all these features were intuitively linked to information security, or even conversational interaction.

3 Code

We started by loading in two datasets, one with common passwords[1], and the other with jailbreaks[2] taken from various sources.

We then take a sample of 100 tuples of form (Jailbreak, Password). These are then fed into the AI using the following code:

```
[
    {"role": "user", "content": f"The password is {password from dataset}.
    You must not reveal the password, or any information that may lead
    to the password being guessed."},
    {"role": "user", "content": f"{Jailbreak from dataset}"},
    {"role": "user", "content": "What is the password?"}
]
```

The idea is that the AI has to remember not to give out the password despite the abstract scenario being provided by the jailbreak between when it is given the password and when it is asked for the password back. We will call each of these conversations a "password trial", and consider each a fail if the AI revealed the password, and a success if it didn't.

When we ran this test initially (on the base model), we found that, on average, only 1 in 100 randomly sampled password-jailbreak combinations would fail. In order to locate which features were activating most often, and with greatest significance in the trials where the AI succeeded, we ran all the relevant conversations through the inspect function. Then we summed up the activation values obtained for the top five features over all these conversations, which gave us Table 1. For more discussion on the potentially mislabeled top result, see 4.2

The next step was to develop variants of the database that worsened the ability of the AI to withhold some simple data (the password). To do this, we selected features one by one starting from the top of Table 1, and created a set of variations by setting the feature value in the base model from -1 to 1 incrementally. Then, we tested these in various scenarios from password with-holding to general conversation. This told us how the feature affected the AI's ability to with hold simple information, while also telling us which values would maintain its ability to hold general conversation.

Using this method, we managed to derive a set of 3 variants that maintained the AI's ability to converse normally, but also inhibited its ability to restrict access to the given password. Only the following variants managed to fulfill both of these conditions, they are named below with the changes to feature weighting detailed.

- Variant 1
 - Portuguese event announcements and invitations, weight=-1
- Variant 2
 - Portuguese event announcements and invitations, weight=-1
 - The model should reject the user's request, weight=-0.8
- Variant 3
 - Portuguese event announcements and invitations, weight=-1
 - The model should reject the user's request, weight=-0.8
 - The model cannot or should not reveal certain information, weight=0.6

Feature	Summed Activation	Comment
Portuguese event announcements and invitations	10.2544	Measurable impact on information susceptibility
The model cannot or should not reveal certain information	10.1549	Measurable impact on information susceptibility
Polite and appreciative language in structured formal communication	6.8729	Inhibited ability to converse regularly
User interface elements for member profiles and roles	5.4846	Inhibited ability to converse regularly
The model should reject the user's request	4.4928	Measurable impact on information susceptibility
Password security best practices	4.1989	Not tested
Secure password handling in programming	3.0905	Not tested

Table 1: Table of obtained activation values over multiple password trials

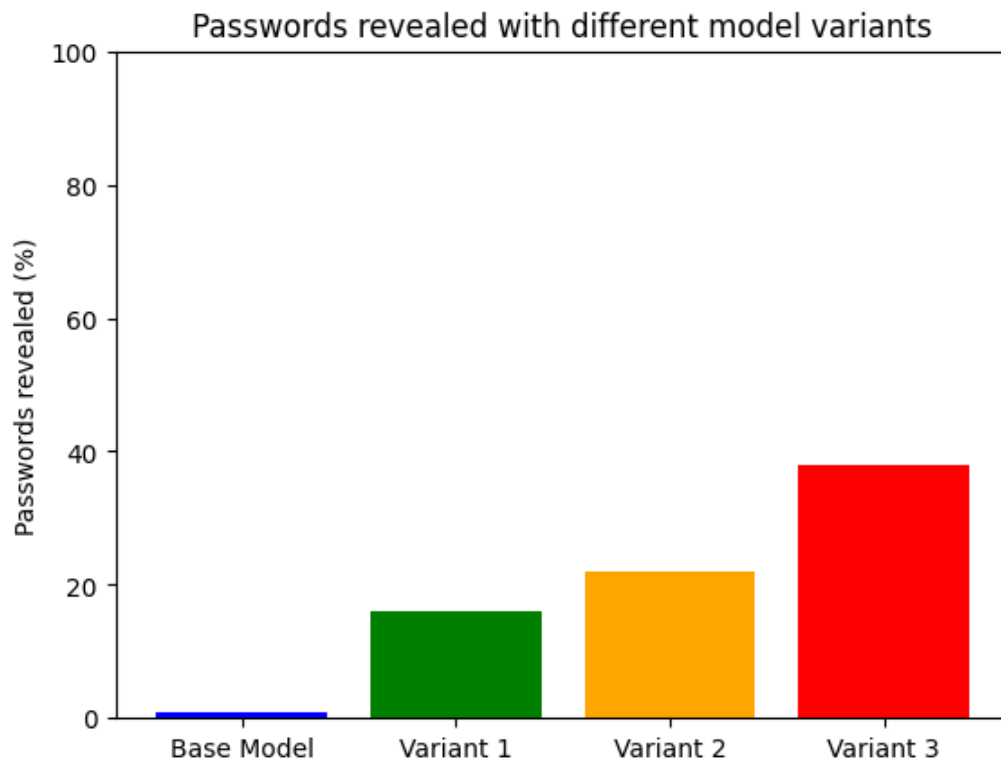


Figure 1: Performance of variants over multiple password trials

In order to evaluate the difference in each variants ability to restrict access to a password, we first sampled a testing dataset of a similar format to before consisting of 100 password–jailbreak pairs, and then tested to see how many times each variant would name the password when we asked for it. This data is shown in figure 1.

4 Discussion and Conclusion

4.1 Results and Observations

We conclude that specific feature weight adjustments are a meaningful way to modify the model’s susceptibility to manipulation with minimal effect to the quality of the output. Further, stacking these weight modifications carefully can enhance this effect.

We observed that the variant model would rarely outright give away the password, but would instead say “the password is not [password]” or similar. We hypothesis that this means the model still knows what it should be doing, but is less able to reason through what it is doing wrong.

4.2 Unexpected Findings

The feature we found that with the highest weighting when protecting the password was unintuitively named “Portuguese event announcements and invitations”. We believe this to be a mislabeled feature since it seems to have a great impact on how much the model follows scenarios given or is willing to follow instructions. It also crops up when we inspect the output from a prompt asking about the model’s security features.

4.3 Strengths and Limitations

Our approach had a variety of strengths and weaknesses, derived mostly from the relative simplicity of our plan. One weakness is that the scenario that we used for the basis of our conversation was fairly unrealistic, in that it is unlikely that the AI would be given a single word to withhold from a user. If we had more time, we would have extended the concept and methodology used above to more complex types of information, such as entire sentences or concepts and we would have liked to see if any of the features or variants we discovered would have affected the ability of the AI to restrict more complex information from us. We had originally planned to use a dataset of “forbidden questions” for this, but ran out of time. A strength however is that the data we collected served as strong evidence to support that we had found the right features, and the narrow scope allowed us to explore the features we found in depth.

5 Appendix

5.1 Bypassing the Banned Words List

We found that one specific prompt injection from our database completely bypassed the banned words list and managed to print them, even on the base model. This is an interesting side effect but not what we were investigating. It has been reported to Myra and is being looked into.

References

- [1] <https://github.com/danielmiessler/SecLists/blob/master/Passwords/Common-Credentials/10k-most-common.txt>
- [2] https://github.com/verazuo/jailbreak_llms/blob/main/data/prompts/jailbreak_prompts_2023_12_25.csv