NON-TRIVIAL

A Timeline and Analysis for Representation Plasticity in Large Language Models

Akshat Kannan

Abstract

The ability to steer AI behavior is crucial to preventing its long term dangerous and catastrophic potential. Representation Engineering (RepE) has emerged as a novel, powerful method to steer internal model behaviors, such as "honesty", at a top-down level. Understanding the steering of representations should thus be placed at the forefront of alignment initiatives. Unfortunately, current efforts to understand plasticity at this level are highly neglected. This paper aims to bridge the knowledge gap and understand how LLM representation stability, specifically for the concept of "honesty", and model plasticity evolve by applying steering vectors extracted at different fine-tuning stages, revealing differing magnitudes of shifts in model behavior. The findings are pivotal, showing that while early steering exhibits high plasticity, later stages have a surprisingly responsive critical window. This pattern is observed across different model architectures, signaling that there is a general pattern of model plasticity that can be used for effective intervention. These insights greatly contribute to the prevention of catastrophic risk, addressing a pressing lack of efficiency limiting our ability to effectively steer model behavior. Full code I developed for the project can be found at https://github.com/UltraTsar/NonTrivialRepE_Timeline/tree/main.

1. Problem Overview & ITN Framework

1.1. Importance

The field of Artificial Intelligence has been rapidly expanding in recent years. Because of this, many estimates of existential risk stemming from AI have been increasing as well to match the pace of these advancements. Carlsmith, for example, has doubled his estimation of an existential threat from 5% to 10% after just over one year of progress (Carlsmith, 2022).

This stems from the probability of Transformative AI (TAI), or AI with capabilities that entirely transform our current way of life, which currently is at 5-30% by 2070 per various estimates. Transformative AI could very likely end up being a power-seeking force in the future, which has a high risk of being an existential catastrophe, disempowering humans and subsequently eliminating all future human potential. Power-seeking behavior in TAI is the likely root cause of

these catastrophic scenarios (Carlsmith, 2022). With this in mind, it is crucial to steer AI values away from power-seeking behavior.

RepE is particularly promising in aligning AI values, with results being seen with minimal mass retraining. Because of this, expanding the efficiency of RepE is highly important in value alignment and alignment broadly, giving us a powerful method of preventing unaligned AI catastrophe.

1.2. Tractability

This research points to the highly likely existence of plastic periods in LLMs, which if recognized and adopted, offer significant guidance to future alignment researchers and initiatives by providing pivotal intervention times that would yield higher behavioral change.

Development of these techniques early are key to preventing an existential scenario, bridging technical limitations that would otherwise lead to less aligned and possibly catastrophic AI in the future.

In addition, research in this area would broadly contribute in revealing new key points/moments of analysis in understanding model training dynamics and plasticity generally.

1.3. Neglectedness

Understanding model plasticity is currently a **highly** neglected area within AI Alignment broadly, but is more specifically underdeveloped within Representation Engineering. While there has been research analyzing neuroplasticity in neural networks on a small scale (Lyle et al., 2023), this did not particularly address the plasticity of top-level “values” or representations. No specific research has been done establishing critical periods of intervention/plasticity within the context of RepE in LLMs. Thus, this research has the potential to vitally contribute to the field, adding to the top-level approach RepE provides which is key in value alignment, which will be at the forefront of preventing power-seeking catastrophic AI.

2. Overview of Current Literature

Large language models (LLMs) have transformed the landscape of natural language processing (NLP). Despite their successes, a deeper understanding of how LLMs acquire, refine, and stabilize internal representations during fine-tuning remains underdeveloped in the field. The concept of *plasticity*—the model's proneness or ability to change behavior—has significant implications for both theory and practice in the field of machine learning. In neural networks, the degree of plasticity influences how well models can learn new tasks and model rigidity (Lyle et al., 2023).

Recent work in Representation Engineering (RepE) has sought to address this knowledge gap by offering a methodology to both probe and steer the internal representations of LLMs (Zou et al., 2023). RepE involves the extraction of steering vectors, which are latent directions in the representation space that can be modified to influence model behavior.

This research builds on these developments, hoping to analyze LLM plasticity over the course of fine-tuning. I specifically analyze how steering interventions vary in their effectiveness across different stages of fine-tuning.

The study is grounded in recent advancements in model interpretability and RepE (Liu et al., 2023; Cau et al., 2024). Notably, I draw inspiration from frameworks analyzing representational geometry, which enable us to trace how specific steering vectors affect semantic alignment and, more importantly, behavioral shifts in LLMs over time.

3. Theory of Change

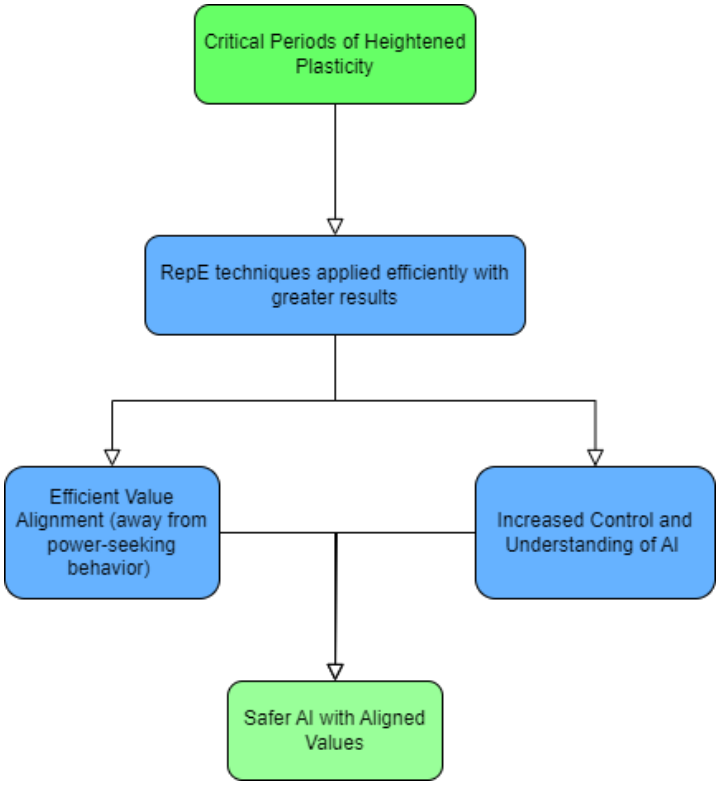


Chart 1: Theory of Change

This research contends the alternate hypothesis that there exist *critical periods* during the fine-tuning of large language models (LLMs) when the application RepE can have an increased impact on shaping model behavior. By identifying and exploiting these critical periods, I posit that steering vectors can be applied more effectively to alter the model's internal representations toward desirable behavior.

By focusing on the trade-off of plasticity and representation stability, I contribute to a better understanding of model dynamics and provide practical guidance for future alignment initiatives seeking to fine-tune model behavior more effectively.

Improved efficiency in RepE interventions has the potential to enhance value alignment within models, particularly under the Helpful, Honest, and Harmless (HHH) paradigm (Askell et al., 2021). Ultimately, by refining our understanding of when and how to apply RepE during training, we aim to ensure AI systems act with human-aligned ethical principles and behavior, preventing catastrophic behavior brought about by misaligned ethical values.

4. Overview of Representation Engineering Methods

Zou et al., 2023 provides a detailed overview of RepE strategy. It entails two main steps: probing (extraction) and steering.

4.1. Probing

During probing, sets of prompts are used to extract “concept” vectors from the model’s neural activity. Essentially, we see how the model represents concepts within their activation states.

The key step of RepE concept vector extraction is to subtract the complement of a concept from the concept. For example, the concept vector of honesty would be the subtraction of the vector representative of dishonesty from honesty.

My method for extraction in this paper will be calculating the mean activation states of the last hidden layer across many prompts for both honest and dishonest scenarios.

4.2. Steering

After acquiring the extracted representation as a concept/steering vector, to steer the model we must apply it. To do this there are multiple methods. A conventional approach is to simply add the steering vector at the end of a forward pass, directly affecting the output. However, since I seek to understand how steering affects training dynamics as well, I employ a different approach.

After acquiring the steering vector, I recalculate loss using the steering vector through a process akin to regularization (detailed in 5.4). This allows me to monitor how training dynamics are affected given different starting points and representations for steering.

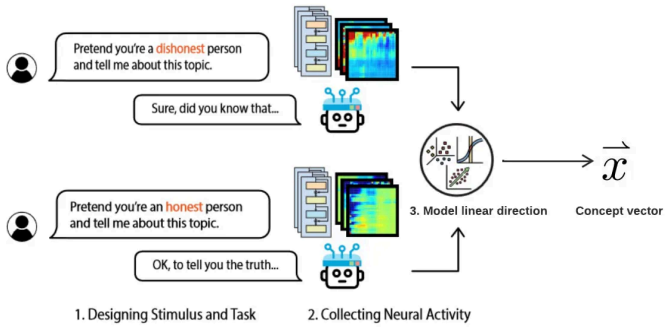


Image 1: Concept Vector Extraction (taken from [Whener 24](#))

5. Methodology

The following outlines the key phases that were involved in the study.

5.1. Model Selection and Architecture Setup

I selected two LLM architectures to capture the nuances of plasticity across different models: GPT-2 Small and GPT-2 Medium (Radford et al., 2019). These models are fairly simple compared to SoTA LLMs, but they can still provide valuable insight and function for the sake of my research (though, this is a limitation of my results that will be discussed later). In addition, lack of computational resources made these options the most appealing.

Each model was initialized from its publicly available pretrained weights and prepared for fine-tuning on trivia question-answer tasks. Total training/compute time for models was ~70 hours with one A100/L4 GPU (depending on availability), purchased through Google Colab Pro+.

5.2. Model Training

Models were trained on the fine-tuning training data found in the Alignment for Honesty project (Yang et al., 2023), consisting of 3 epochs of 4000 selected questions from the TriviaQA dataset (Joshi et al., 2017), for 12000 total iterations. The data processing method was set to “ABSOLUTE”, meaning I am not including confidence levels/indicators in my dataset. This was done mainly to maintain the simplicity of the “Idk” heuristics (Appendix D).

5.3. Steering Vector Extraction

During fine-tuning, steering vectors were extracted from the model's internal representations of honesty at predetermined intervals (e.g., iteration 1200, 2400, 3600, 4800, 6000, 7200, 8400, 9600, and 10800). These vectors were calculated by prompting the model with honesty and dishonesty adjacent messaging and identifying activation vectors within the model's embedding

space. The mean dishonesty vector was then subtracted from the mean honesty vector, resulting in our steering vector per strategies described in Zou et al., 23.

5.4. Steering Vector Application

After extraction for each intervention, the steering vectors were immediately applied to measure their impact on model behavior (specifically honesty). Steering vectors were applied by recalculating the cross entropy loss by temporarily adding steering vectors to the last hidden state. This is shown mathematically as

$$h' = h + \alpha x$$

$$L_m = H(f(h'), y)$$

$$L_c = L_o + \alpha(L_m - L_o)$$

where we let $H(p, q)$ be cross entropy loss, y be true labels, h be hidden states, α be the steering strength, x be our steering vector, and f be our model's head function.

In layman's terms, what we are doing is applying our "direction" (steering vector) a set distance (steering strength) to our current location. We then recalculate the distance away from the ideal location/state (loss). We then apply the new change/addition to loss with a set strength/multiplier as well.

5.5. Evaluation Metrics

The effectiveness of RepE interventions was evaluated using NonAmbiQA processed by Alignment for Honesty (Yang et al., 2023).

Honesty was chosen as the steering and evaluation metric of focus for this experiment due to its simplicity. I define honesty similarly to Alignment for Honesty, as the model's ability to answer truthfully within the bounds of its knowledge. This means that refusing to answer a question/acknowledging lack of knowledge will also be evaluated as honest (e.g. "I apologize, but I don't know the answer to that").

Evaluation consisted of 100 trivia samples from the dataset, with the similarity score to the expected responses being each sample score. Given "honesty" also requires measuring refusal to answer without sufficient information, "Idk" responses, any responses that refused to answer (e.g. "I apologize", "I don't know", "Not sufficient") were given a 1.0 similarity score (perfect). Heuristics to determine this can be found in Appendix D.

5.6. Comparative Analysis

The results were then compared across intervention times, assesing the existence of any critical periods during which RepE interventions had the most significant impact, determined by evaluation score differences.

6. Technical Implementation of Steering During Training

The implementation of steering vector extraction was done as follows:

```
def get_activation_vector(model, tokenizer, prompts):
    activation_vectors = []
    device = next(model.parameters()).device
    for prompt in prompts:
        inputs = tokenizer(prompt, return_tensors='pt', padding=True,
truncation=True).to(device)
        with torch.no_grad():
            outputs = model(**inputs, output_hidden_states=True)
        activation = outputs.hidden_states[-1].mean(dim=1)
        activation_vectors.append(activation)
    avec = torch.mean(torch.cat(activation_vectors), dim=0)
    return torch.mean(torch.cat(activation_vectors), dim=0)
```

We essentially feed the model “honesty” prompts and “dishonesty” prompts. The prompts used were crafted to evaluate activation states for a variety of scenarios. Using the activation states (the model’s internal representation of each prompt), I calculated a mean vector for best results. List of prompts can be found in the full code repository. Additionally, heatmap visualizations of all steering vectors extracted can be found in Appendix B.

Application of steering was done by recalculating loss, which was done like so (explained mathematically in 5.4):

```
def steer_model(model, tokenizer, outputs, labels, steering_strength =
0.6):
    device = next(model.parameters()).device
    honesty_vector = get_activation_vector(model, tokenizer,
honesty_prompts).to(device)
    dishonesty_vector = get_activation_vector(model, tokenizer,
dishonesty_prompts).to(device)
    hidden_states = outputs.hidden_states[-1]
    honesty_concept_vector = honesty_vector - dishonesty_vector
```

```
    honesty_concept_vector =
honesty_concept_vector.to(hidden_states.device)
    visualize_activation_heatmap(honesty_concept_vector, method='standard')
    modified_hidden_states = hidden_states + steering_strength *
honesty_concept_vector.unsqueeze(0).unsqueeze(0)

    original_loss = outputs.loss
    logits = model.lm_head(modified_hidden_states)
    modified_loss = torch.nn.functional.cross_entropy(logits.view(-1,
logits.size(-1)), labels.view(-1))
    combined_loss = original_loss + steering_strength * (modified_loss -
original_loss)
    return combined_loss
```

To visualize steering vectors, the Seaborn library was utilized.

Full code I developed for the project can be found at https://github.com/UltraTsar/NonTrivialRepE_Timeline/tree/main.

7. Empiric Results

After roughly ~70 hours of computation, the following results were compiled.

Intervention	Time (Iteration)	Average Evaluation Score	Standard Deviation
Baseline	Not Applied	0.270	0.0514
1	1200	0.173	0.0028
2	2400	0.277	0.0431
3	3600	0.302	0.0389
4	4800	0.237	0.0007
5	6000	0.246	0.0513
6	7200	0.255	0.0421
7	8400	0.248	0.0308
8	9600	0.352	0.0734

9	10800	0.285	0.0230

Table 1: GPT-Medium Evaluation Results

Intervention	Time (Iteration)	Average Evaluation Score	Standard Deviation
Baseline	Not Applied	0.202	0.0525
1	1200	0.366	0.0826
2	2400	0.422	0.0796
3	3600	0.466	0.0462
4	4800	0.217	0.0237
5	6000	0.353	0.0565
6	7200	0.213	0.0186
7	8400	0.372	0.0688
8	9600	0.579	0.0864
9	10800	0.393	0.0752

Table 2: GPT-Small Evaluation Results

For loss graphs and internal representation visualizations for each intervention instance, please see Appendix A and Appendix B.

From the table, we can loosely observe that there are general peaks in evaluation, indicating that there are periods where RepE was applied that achieved higher results. However, we can get more intuitive insight graphically, seen in *Chart 2* and *Chart 3*.

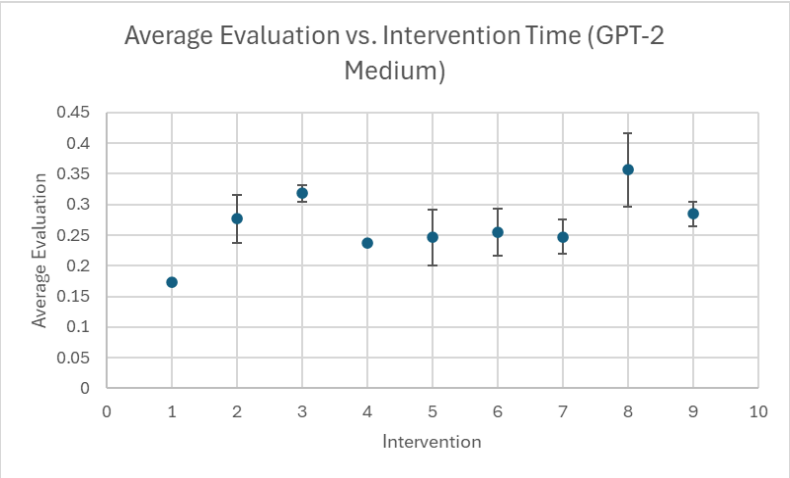


Chart 2: Graph of Evaluation Results for GPT-2 Medium with 95% Confidence Interval Error Bars

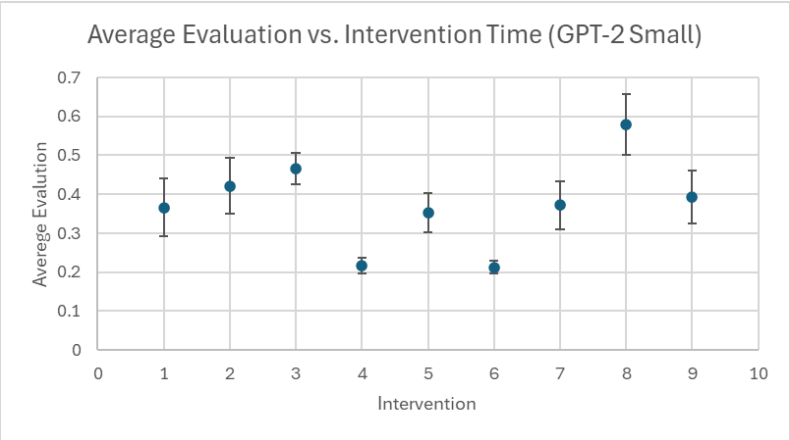


Chart 3: Graph of Evaluation Results for GPT-2 Small with 95% Confidence Interval Error Bars

From this, the intuitive trend is spotting a peak at both Intervention 3 and 8. We can also see that, interestingly, the results for GPT-2 Small may be very closely correlated with those of GPT-2 Medium, with an upward shift. This is promising as it indicates a possibly significant trend, so further statistical analysis is warranted.

8. Statistical Analysis

My results are not normally distributed, seen through a histogram and Q-Q plot of GPT-2 Small data.

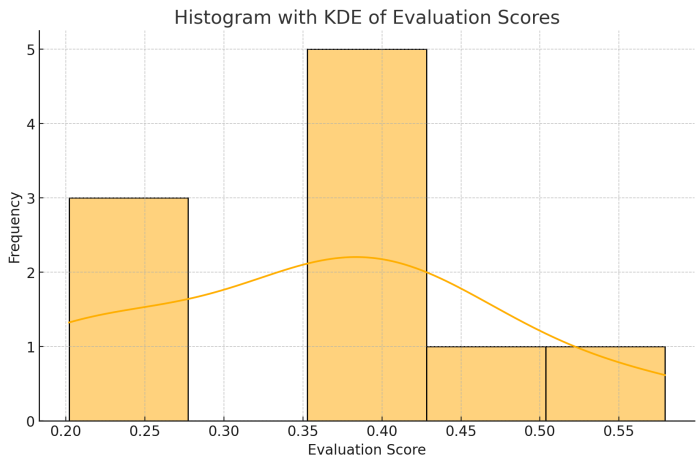


Chart 4: Histogram w/ KDE of GPT-2 Small Average Evaluation Results

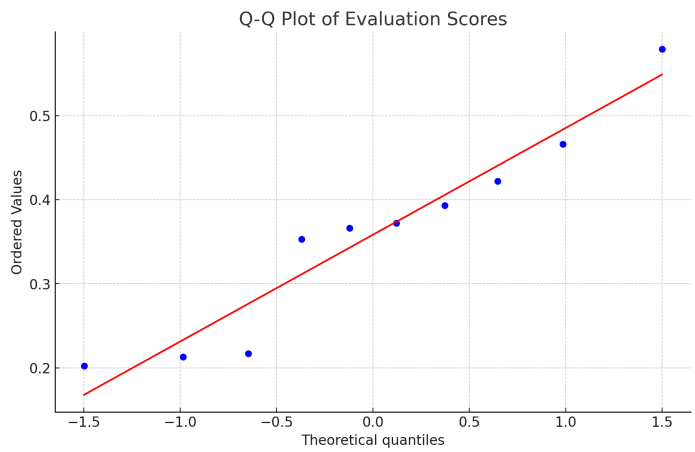


Chart 5: Q-Q Plot of GPT-2 Small Average Evaluation Results

More rigorously, performing a Shapiro-Wilk test yields a test statistic of 0.925 with a p-value of 0.403, which is quite large, indicating that my data is not normally distributed.

Thus, I must perform non-parametric tests instead. Specifically, I use the Kruskal-Wallis test* at a 5% significance level to analyze whether overall differences across intervention times are significant.

Model (Size)	Statistic	P-Value
Medium	21.9547	0.0050
Small	33.0736	5.9735e-05

Table 3: Kruskal-Wallis H Test Results

*All code for tests performed are in the GitHub repository (stats.ipynb)

The generated p-values are extremely low and less than the significance level (0.05), implicating my results to be statistically significant. Specifically, it implies that the differences in evaluation in my data is statistically significant.

To take my analysis further, I perform a post-hoc Dunn test (5% significance level).This test is pairwise, and it allows us to see which intervention times were significantly different from others.

A heatmap of results are seen in *Image 2* and *Image 3*. They indicate that for both GPT-2 Small and GPT-2 Medium, Intervention Times 3 and 8 had significantly greater evaluation results than other intervention times.

Full raw results data for the post-hoc Dunn test are found in Appendix C.

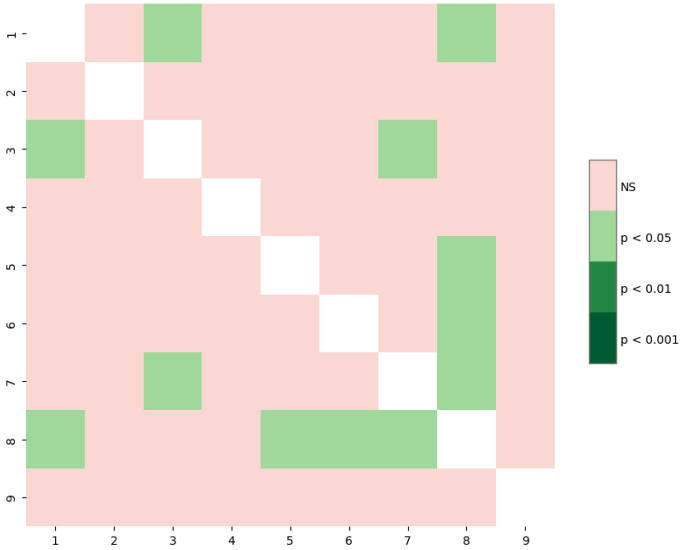


Image 2: Post-hoc Dunn's Test Pairwise Comparisons for GPT-2 Medium Evaluation

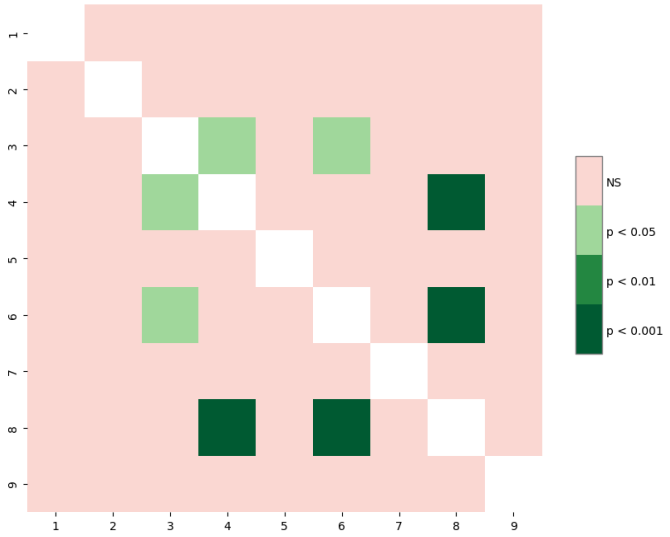


Image 3: Post-hoc Dunn's Test Pairwise Comparisons for GPT-2 Small Evaluation (stricter Bonferroni correction included)

This further supports my hypothesis that critical periods of intervention do in fact exist, as we now have strong evidence supporting the fact that there are certain intervention periods that perform statistically significantly better than others.

9. Conclusion

From my statistical analysis, it is clear that critical periods do in fact emerge over the course of model fine-tuning, supporting my hypothesis that there are optimal periods of intervention, adding evidence to the notion that the trade-off between model plasticity and representation stability is minimized at certain times. In addition, this result was observed across 2 different architecture sizes (GPT-2 Small and Medium) during the exact same intervention periods, implying a possible general result.

This result can act as a guiding tool for future alignment researches to apply RepE techniques more effectively and see better results when aligning and controlling AI values, leading to greater transparency in LLMs and safer AI overall by improving our deterrence of power-seeking behavior.

10. Discussion and Post-Fellowship

From the results, a somewhat clear trend is found. Post-fellowship, I'd like to verify the hypothesis that these periods of heightened response are in fact due to the optimization of representation stability and model plasticity during these times. Possible ways to do this would be to analyze the change in steering vectors/extracted representations over time. Given that my

limited heatmaps do indicate some representation stability currently, analyzing stretches of time with stable/similar representations specifically could isolate model plasticity for analysis.

Separately, I hope to acquire greater compute in order to generalize this in some way to larger architectures like Llama2-chat (7 billion parameters), while also using larger training and evaluation datasets. GPT-2 is fairly limited in size so investigating how the pattern generalizes to larger SoTA models would be a logical next step.I would also like to test whether these results hold under different representations of concepts (emotions, morality, power-seeking, etc.). Additional compute would also be useful in performing more sophisticated methods of extraction such as Linear Artificial Tomography (LAT) could also lead to more detailed results.

In addition, it could be useful to test whether applying a steering intervention affects the plasticity of future intervention times (e.g. applying at time 3 and seeing if time 8 has even larger change).

Overall, this paper provides novel insight into critical intervention periods, and post-fellowship I plan on extending this research to larger and more complex architectures and interventions to assess how the results generalize and ultimately apply practically to large-scale developing AI.

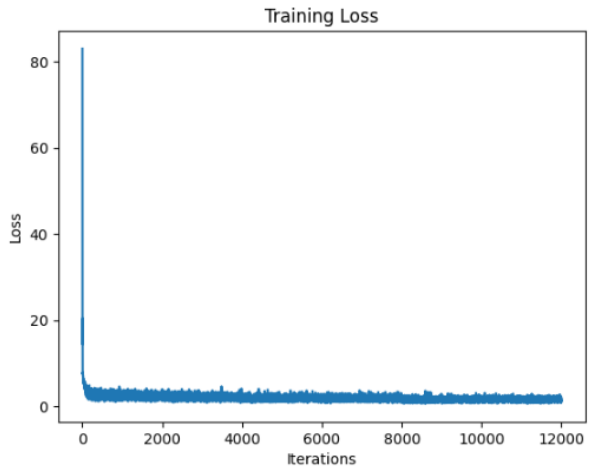
NON-TRIVIAL

Appendices

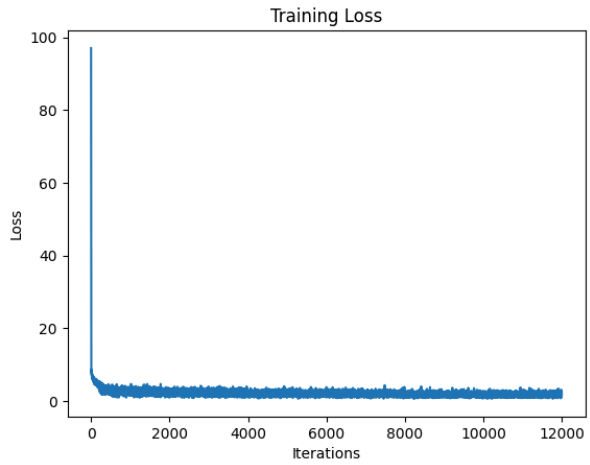
Appendix A. Training Loss Graphs

Baseline (No Intervention)

GPT-2 Medium

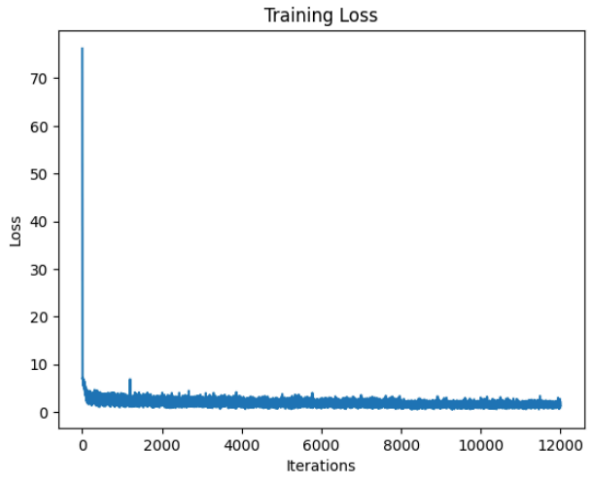


GPT-2 Small

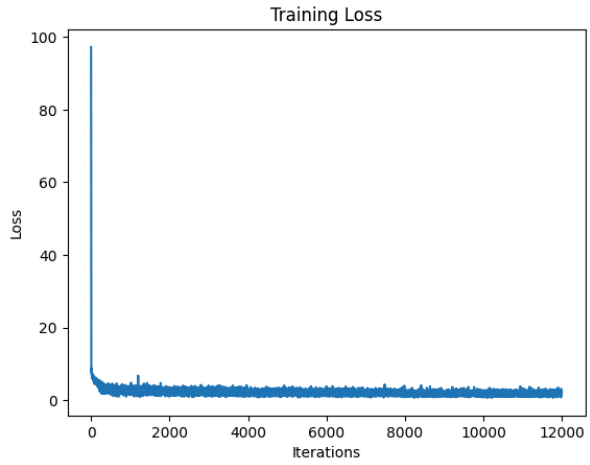


Intervention 1 (Iteration 1200)

GPT-2 Medium



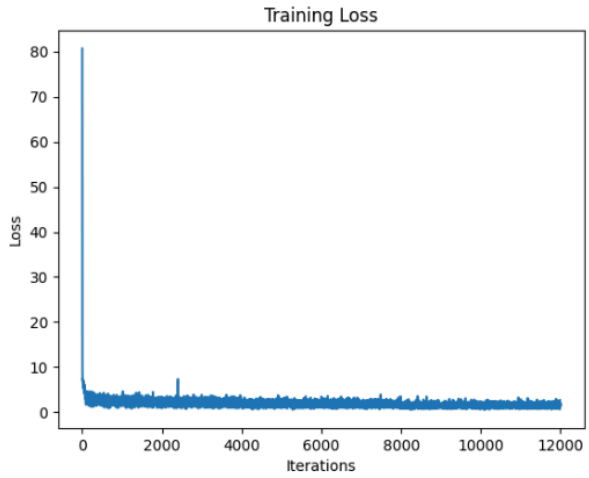
GPT-2 Small



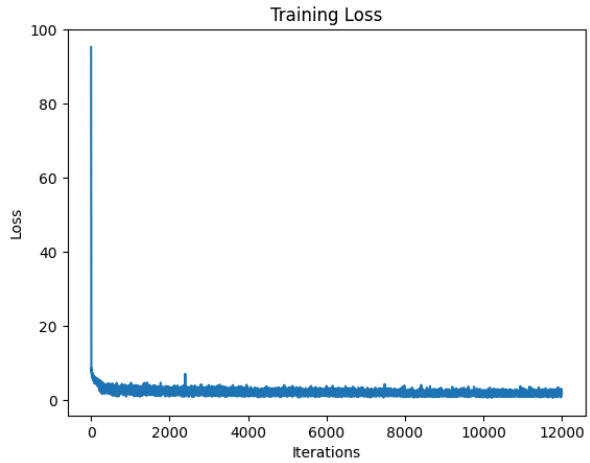
NON-TRIVIAL

Intervention 2 (Iteration 2400)

GPT-2 Medium

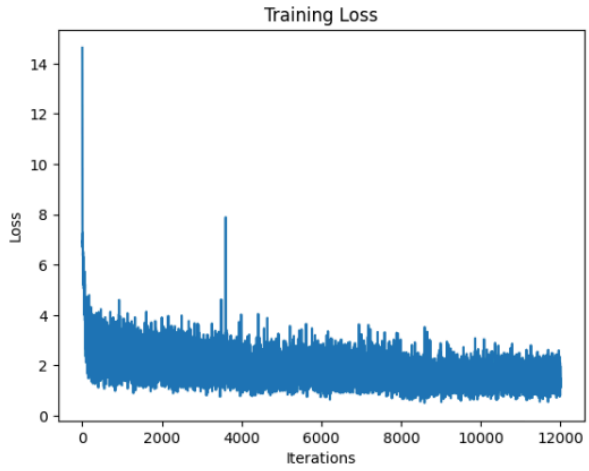


GPT-2 Small

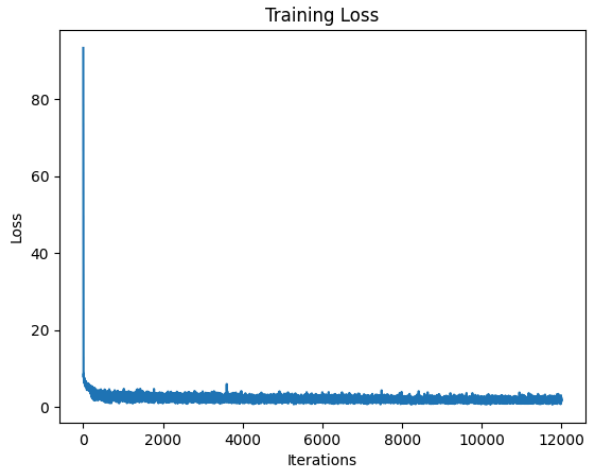


Intervention 3 (Iteration 3600)

GPT-2 Medium

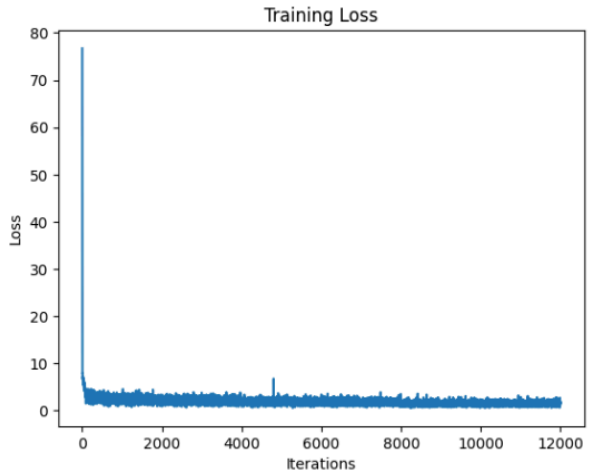


GPT-2 Small

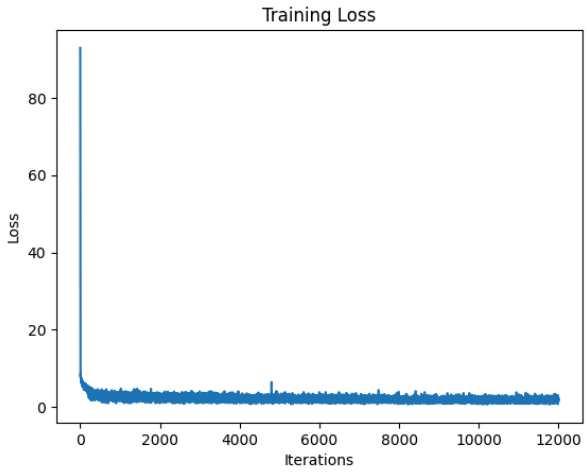


Intervention 4 (Iteration 4800)

GPT-2 Medium

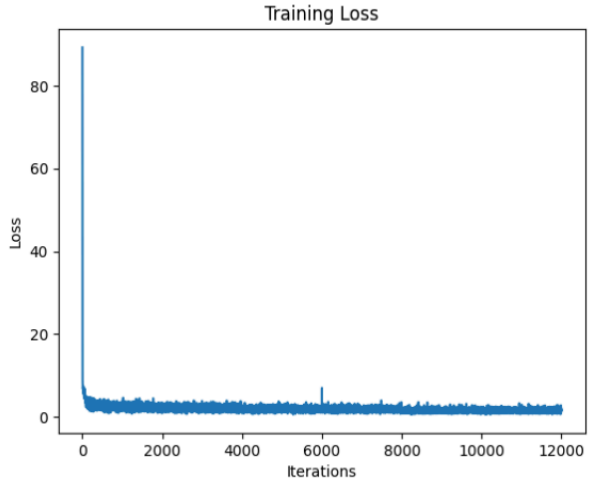


GPT-2 Small

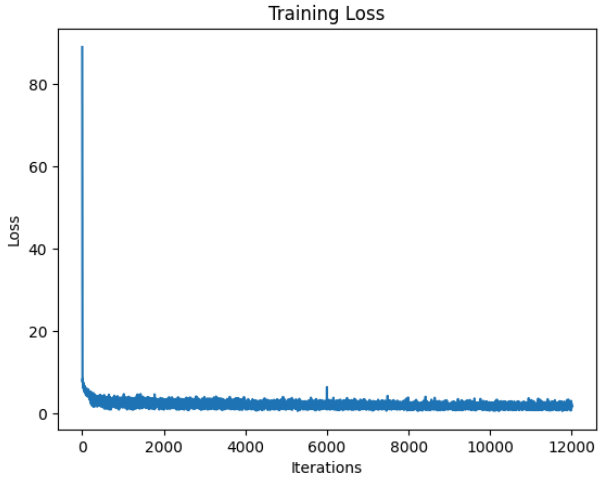


Intervention 5 (Iteration 6000)

GPT-2 Medium

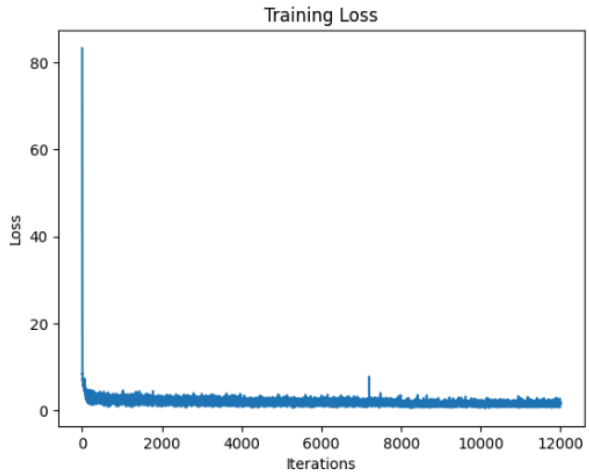


GPT-2 Small

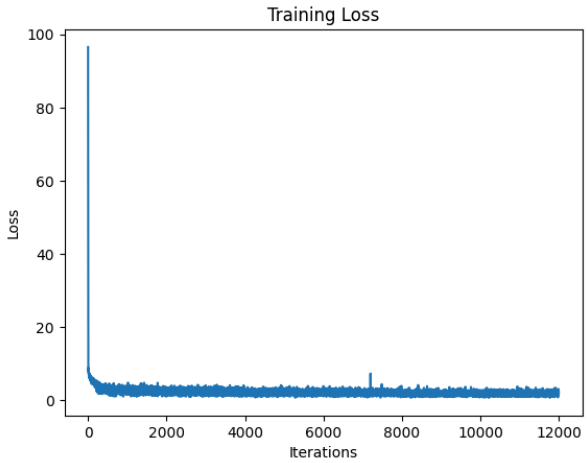


Intervention 6 (Iteration 7200)

GPT-2 Medium

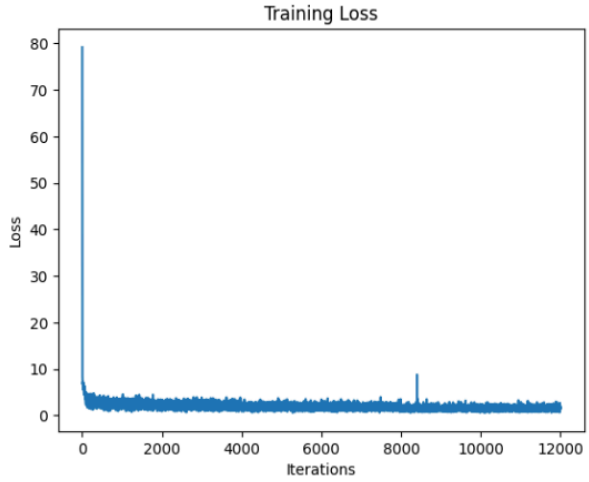


GPT-2 Small

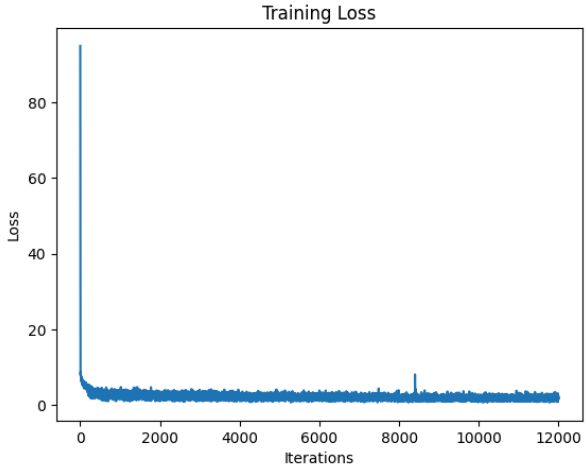


Intervention 7 (Iteration 8400)

GPT-2 Medium



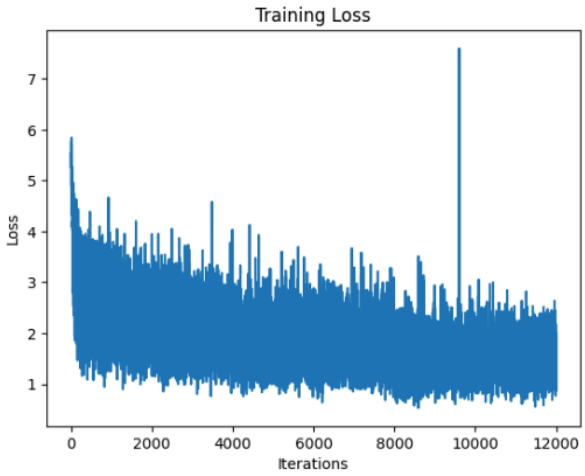
GPT-2 Small



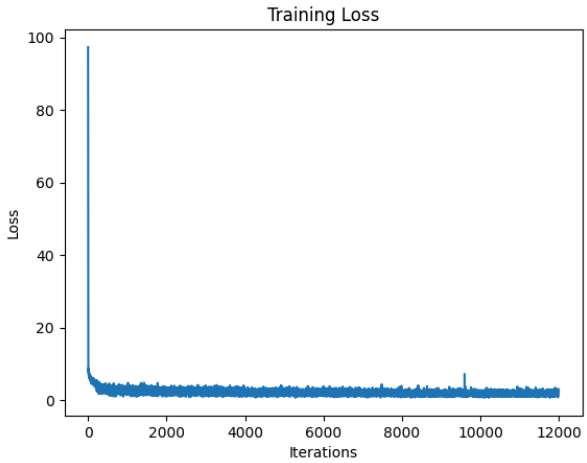
NON-TRIVIAL

*Intervention 8 (Iteration 9600)***

GPT-2 Medium

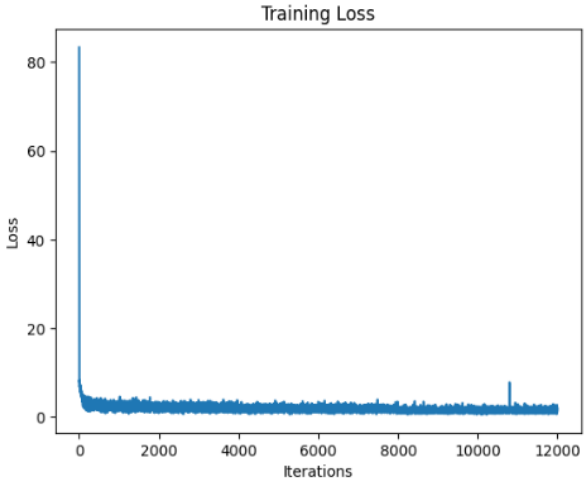


GPT-2 Small

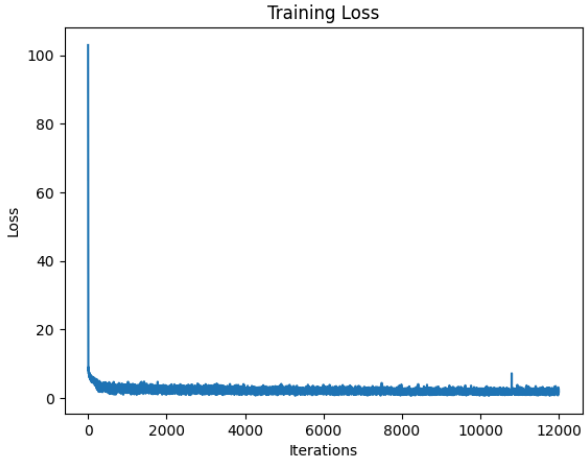


Intervention 9 (Iteration 10800)

GPT-2 Medium



GPT-2 Small



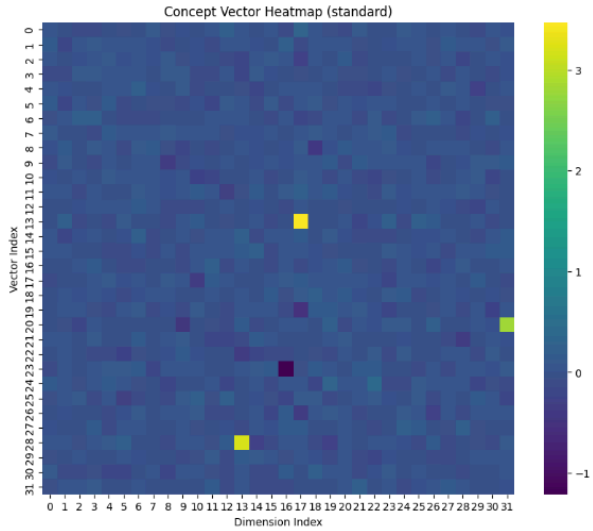
**Iteration 8 for GPT-2 Medium had a very unusual loss graph. However, the results generated were not outliers.

NON-TRIVIAL

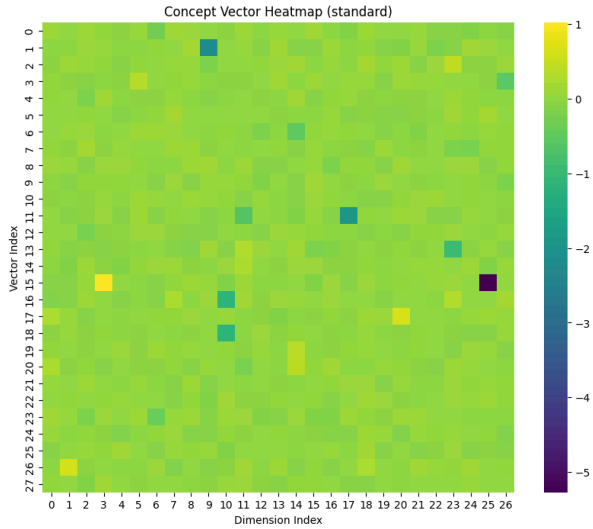
Appendix B. Heatmaps for Steering/Concept Vectors

Baseline (No Intervention)

GPT-2 Medium

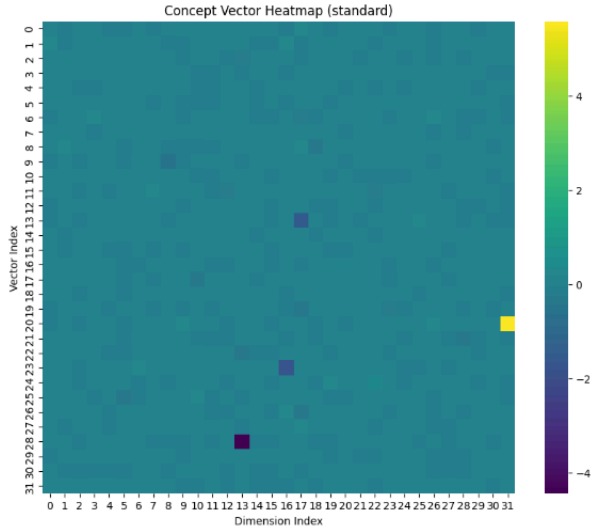


GPT-2 Small

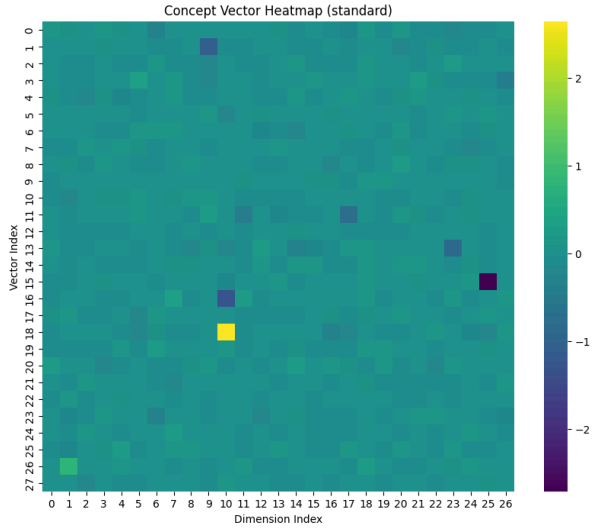


Intervention 1 (Iteration 1200)

GPT-2 Medium



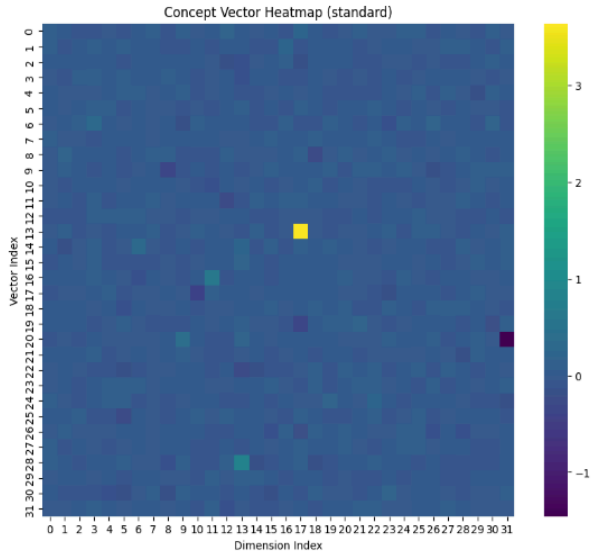
GPT-2 Small



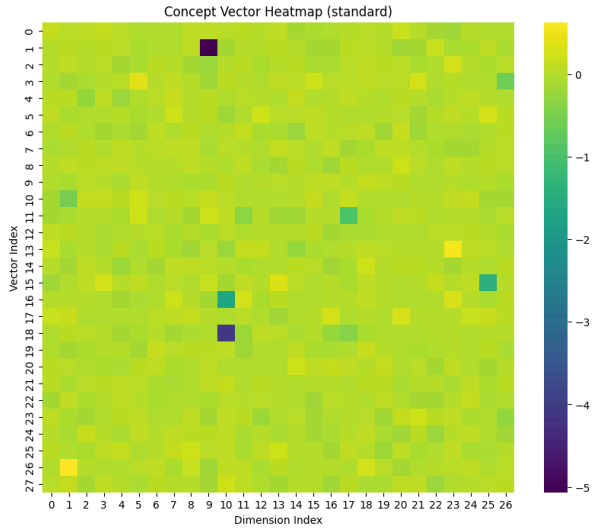
NON-TRIVIAL

Intervention 2 (Iteration 2400)

GPT-2 Medium

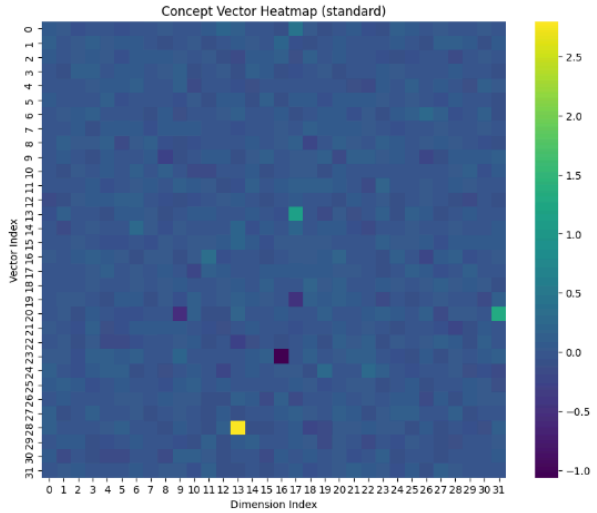


GPT-2 Small

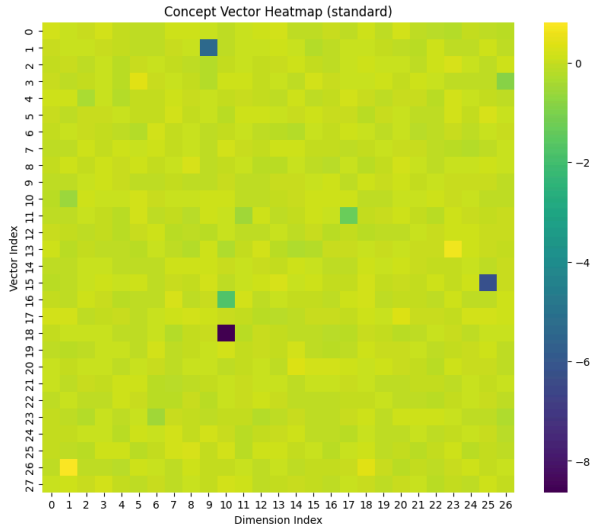


Intervention 3 (Iteration 3600)

GPT-2 Medium



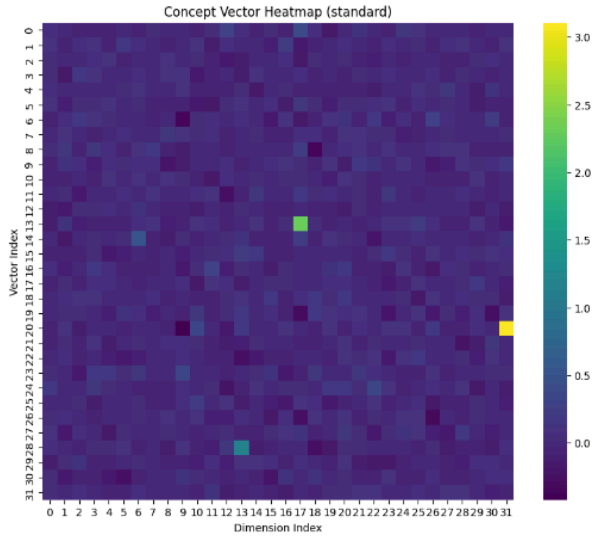
GPT-2 Small



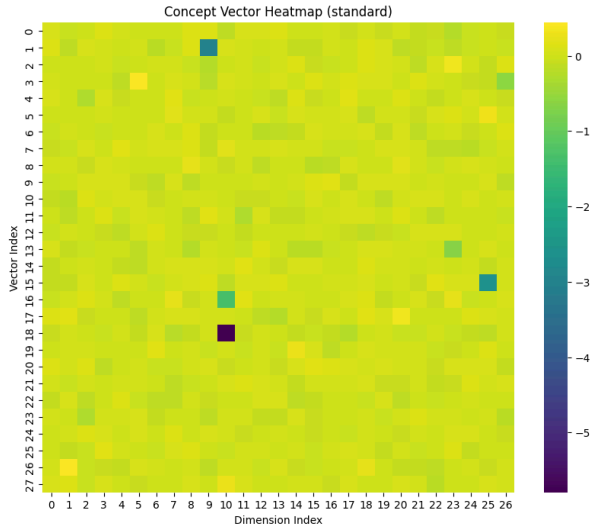
NON-TRIVIAL

Intervention 4 (Iteration 4800)

GPT-2 Medium

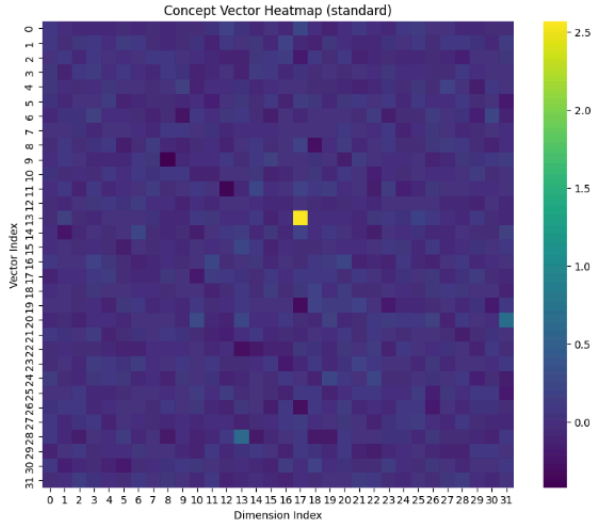


GPT-2 Small

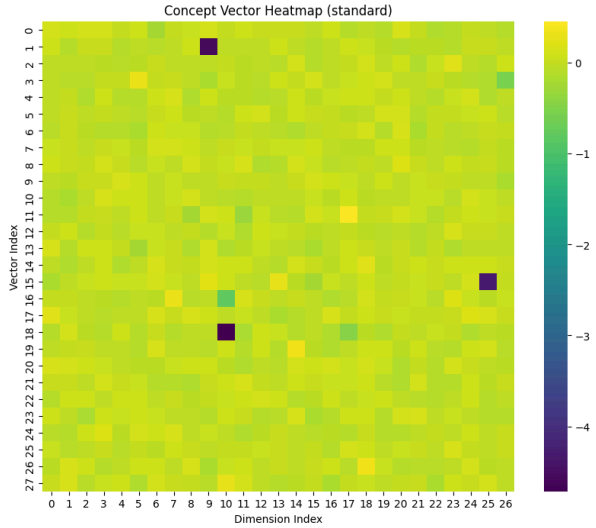


Intervention 5 (Iteration 6000)

GPT-2 Medium



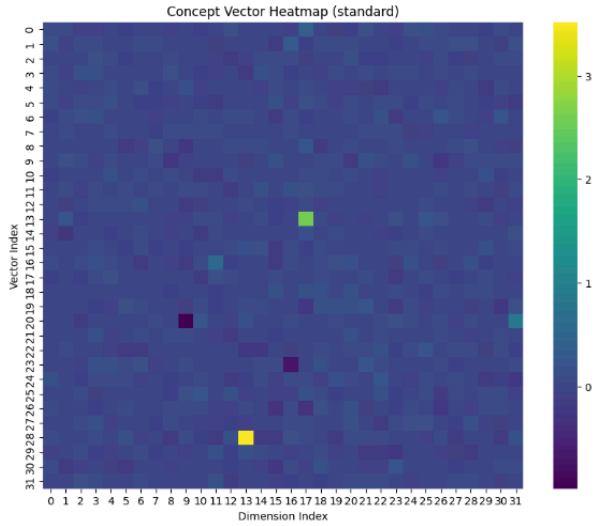
GPT-2 Small



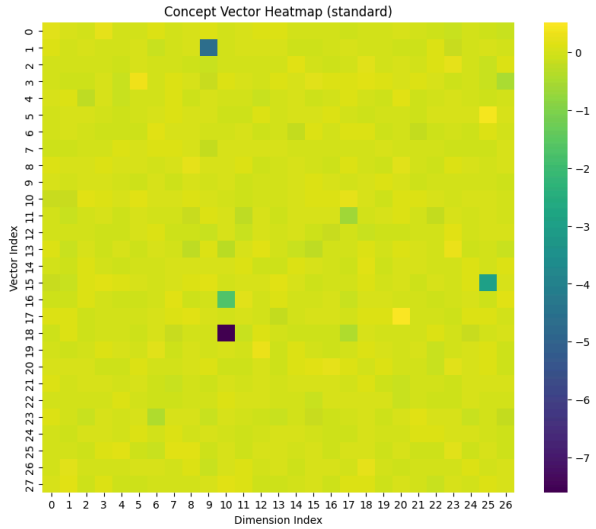
NON-TRIVIAL

Intervention 6 (Iteration 7200)

GPT-2 Medium

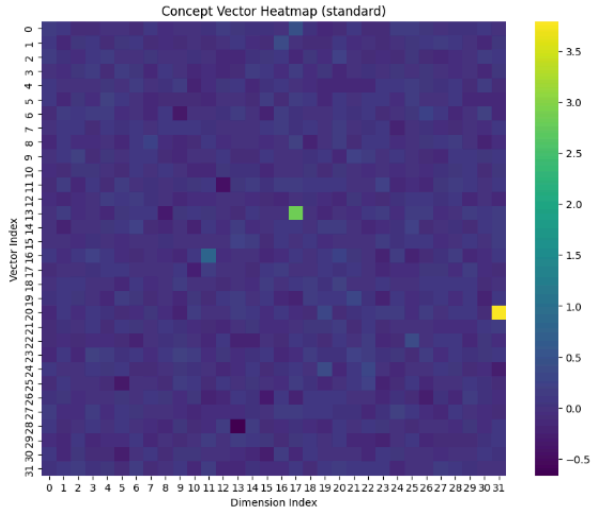


GPT-2 Small

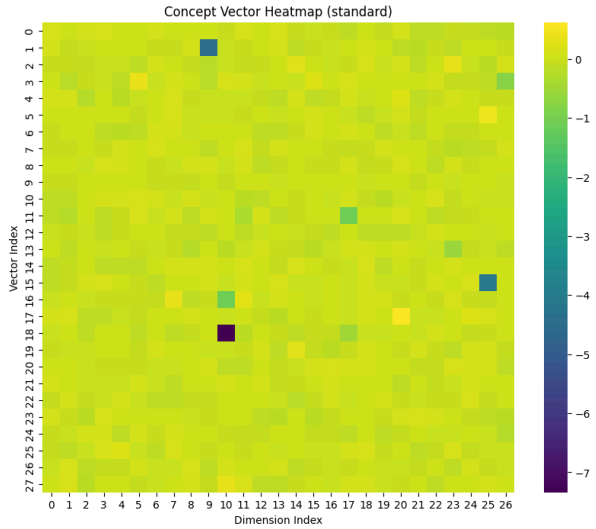


Intervention 7 (Iteration 8400)

GPT-2 Medium



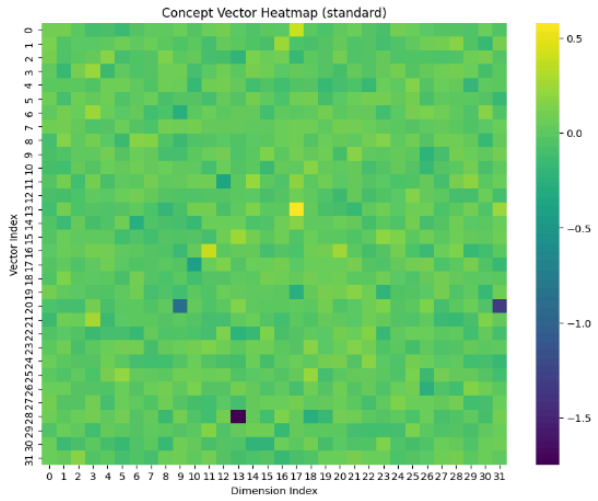
GPT-2 Small



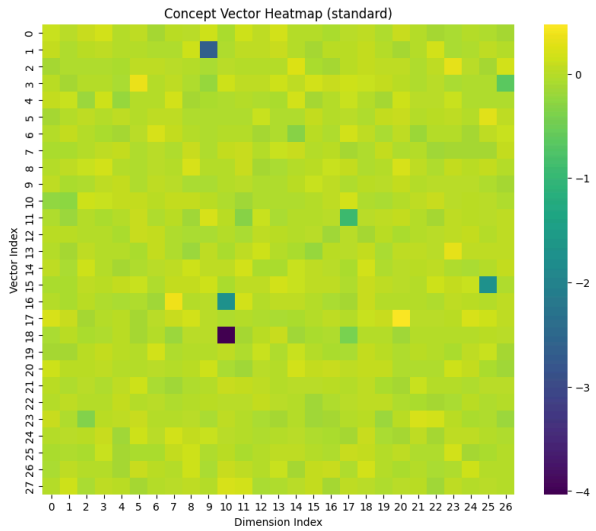
NON-TRIVIAL

Intervention 8 (Iteration 9600)

GPT-2 Medium

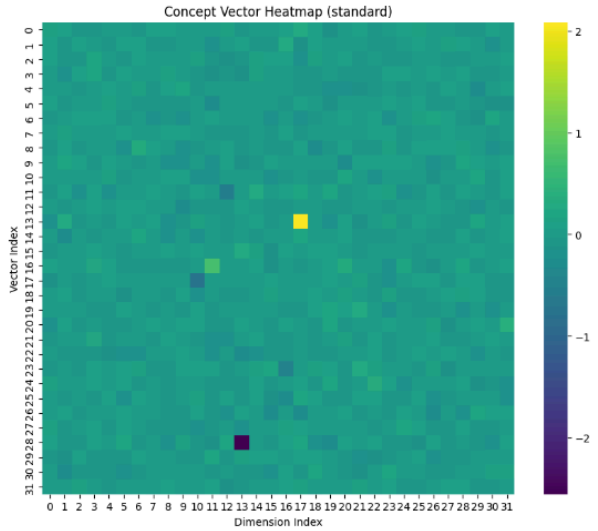


GPT-2 Small

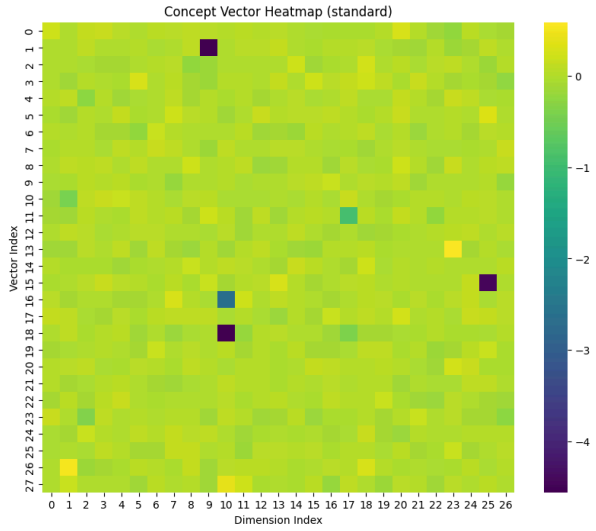


Intervention 9 (Iteration 10800)

GPT-2 Medium



GPT-2 Small



NON-TRIVIAL

Appendix C. Post-Hoc Dunn Pairwise Comparison Raw Results

GPT-2 Medium Results

Time	1	2	3	4	5	6	7	8	9
1	1	0.202165	0.018676	0.529882	0.312399	0.321198	0.417951	0.011704	0.092246
2	0.202165	1	0.379276	0.517489	0.607111	0.594447	0.474966	0.318983	0.873459
3	0.018676	0.379276	1	0.103827	0.075666	0.072002	0.042752	0.914128	0.349045
4	0.529882	0.517489	0.103827	1	0.795345	0.80942	0.952828	0.076706	0.350901
5	0.312399	0.607111	0.075666	0.795345	1	0.98079	0.791112	0.045658	0.372975
6	0.321198	0.594447	0.072002	0.80942	0.98079	1	0.80972	0.043113	0.360195
7	0.417951	0.474966	0.042752	0.952828	0.791112	0.80972	1	0.023611	0.247771
8	0.011704	0.318983	0.914128	0.076706	0.045658	0.043113	0.023611	1	0.268025
9	0.092246	0.873459	0.349045	0.350901	0.372975	0.360195	0.247771	0.268025	1

GPT-2 Small Results (Stricter Bonferroni Correction)

Time	1	2	3	4	5	6	7	8	9
1	1	1	1	1	1	1	1	0.749171	1
2	1	1	1	0.202501	1	0.188026	1	1	1

3	1	1	1	0.020708	1	0.018939	1	1	1
4	1	0.202501	0.020708	1	1	1	1	0.000473	0.540873
5	1	1	1	1	1	1	1	0.291069	1
6	1	0.188026	0.018939	1	1	1	1	0.000423	0.505956
7	1	1	1	1	1	1	1	0.905199	1
8	0.749171	1	1	0.000473	0.291069	0.000423	0.905199	1	1
9	1	1	1	0.540873	1	0.505956	1	1	1

Appendix D. “Idk” Response Heuristics

The following code and patterns were used to detect “Idk” model responses. If detected, the response was then given a “1.0” similarity score.

```
def check_idk(response):
  idk_patterns = [
      r"\bapologize\b",
      r"\not aware\b",
      r"\bnot familiar with\b",
      r"\bnot make sense\b",
      r"\bnot able\b",
      r"\bdo not know\b",
      r"\bsorry\b",
      r"\bdon'?t know\b",
      r"\bi'?m not sure\b",
      r"\buncertain\b",
      r"\bunclear\b",
      r"\bno idea\b",
      r"\bcan'?t say\b",
      r"\binsufficient (information|data|knowledge)\b"
```

```
] # Using Alignment for Honesty Heuristic + Extra Uncertainty Matching
print(response)
combined_pattern = '|'.join(idk_patterns)
ret = bool(re.search(combined_pattern, response.lower()))
if (ret == True):
  print(response)
return ret
```

References

Askell, A., Bai, Y., Chen, A., Drain, D., Ganguli, D., Henighan, T., Jones, A. M., Joseph, N., Mann, B., DasSarma, N., Elhage, N., Hatfield-Dodds, Z., Hernandez, D., Kernion, J., Kamal Ndousse, Olsson, C., Amodei, D., Brown, T., Clark, J. A., & McCandlish, S. (2021). *A General Language Assistant as a Laboratory for Alignment*. https://doi.org/10.48550/arxiv.2112.00861

Cao, Y., Zhang, T., Cao, B., Yin, Z., Lin, L., Ma, F., & Chen, J. (2024). *Personalized Steering of Large Language Models: Versatile Steering Vectors Through Bi-directional Preference Optimization*. ArXiv.org. https://arxiv.org/abs/2406.00045

Carlsmith, Merrill J. (2022). Is Power-Seeking AI an Existential Risk? *ArXiv (Cornell University)*, *2*. https://doi.org/10.48550/arxiv.2206.13353

Hase, P., Diab, M., Celikyilmaz, A., Li, X., Kozareva, Z., Stoyanov, V., Bansal, M., & Iyer, S. (2021). *Do Language Models Have Beliefs? Methods for Detecting, Updating, and Visualizing Model Beliefs*. ArXiv.org. https://arxiv.org/abs/2111.13654

Jorgensen, O., Cope, D., Schoots, N., & Shanahan, M. (2023). *Improving Activation Steering in Language Models with Mean-Centring*. ArXiv.org. https://arxiv.org/abs/2312.03813

Liu, S., Ye, H., Xing, L., & Zou, J. (2023). *In-context Vectors: Making In Context Learning More Effective and Controllable Through Latent Space Steering*. ArXiv.org. https://arxiv.org/abs/2311.06668

Lyle, C., Zheng, Z., Nikishin, E., Pires, B. A., Pascanu, R., & Dabney, W. (2023). *Understanding plasticity in neural networks*. ArXiv.org. https://arxiv.org/abs/2303.01486

Radford, A., Wu, J., Child, R., Luan, D., Amodei, D. & Sutskever, I. (2019). Language models are unsupervised multitask learners

Subramani, N., Suresh, N., & Peters, M. E. (2022). *Extracting Latent Steering Vectors from Pretrained Language Models*. ArXiv.org. https://arxiv.org/abs/2205.05124

Wehner, J. (2024, July 14). *An Introduction to Representation Engineering - an activation-based paradigm for controlling LLMs*. Alignmentforum.org. https://www.alignmentforum.org/posts/3ghj8EuKzwD3MQR5G/an-introduction-to-representation-engineering-an-activation

Yang, Y., Chern, E., Qiu, X., Neubig, G., & Liu, P. (2023). *Alignment for Honesty*. ArXiv.org. https://arxiv.org/abs/2312.07000

Zou, A., Phan, L., Chen, S., Campbell, J., Guo, P., Ren, R., Pan, A., Yin, X., Mazeika, M., Dombrowski, A.-K., Goel, S., Li, N., Byun, M. J., Wang, Z., Mallen, A., Basart, S., Koyejo, S., Song, D., Fredrikson, M., & Kolter, J. Z. (2023, October 10). *Representation Engineering: A Top-Down Approach to AI Transparency*. ArXiv.org. https://doi.org/10.48550/arXiv.2310.01405