Steering Large Language Models using Conceptors: An Alternative to Point-Based Activation Engineering

Bachelor's Project Thesis

Joris Postmus, s4240162, j.postmus@student.rug.nl,
Supervisors: Steven Abreu, s.abreu@rug.nl
& Dr. Herbert Jaeger, h.jaeger@rug.nl

Abstract: While large language models (LLMs) have revolutionized the field of artificial intelligence, reliably controlling their outputs remains a pressing challenge. This project aims to improve a proposed technique called activation engineering where the outputs of pre-trained LLMs are controlled by directly manipulating the models' activations at inference time. Traditionally, this manipulation involves the addition of a steering vector onto the models' activations at a specific stage in the processing pipeline. In contrast to representing the steering target as a single point (vector) in high-dimensional space, we explore the use of conceptors, mathematical objects that can represent a set of activation vectors as ellipsoidal regions in their high-dimensional space. For steering purposes, we use conceptors as (soft) projection matrices that can tune a given activation vector toward a steering target. Due to the aforementioned properties, we hypothesize that conceptors provide more precise control over capturing and steering toward complex activational representations compared to point-based methods. Our experiments show that compared to traditional point-based steering methods, conceptors, especially when combined with a performance enhancement called mean-centering, achieve higher accuracy across multiple steering tasks. These findings suggest that conceptors are a promising tool for effectively controlling the outputs of LLMs, paving the way for further research to fully establish their utility.

1 Introduction

1.1 Motivation

Over the past few years, large language models (LLMs) have caused a major disruption in the field of artificial intelligence (AI). Leveraging a large number of parameters, novel attention mechanisms (Vaswani et al., 2023), and trained on massive corpora, these models have caused a significant leap in AI capabilities (Xu & Poo, 2023). In a short period of time, LLMs have rapidly transformed the focus of industry and academia, and are on track to greatly influence future technological advancements and global societal progress (Zhao et al., 2023; Bubeck et al., 2023; Chang et al., 2023).

The rise of LLMs also presents significant challenges, including the spread of misinformation (Pan et al., 2023), the propagation of social biases (Gallegos et al., 2024), and the emergence of potentially dangerous capabilities (Shevlane et al., 2023). Therefore, as progress on LLMs continues to advance, understanding their internal workings and developing methods to steer these models toward desired (or away from undesired) behaviors is becoming an increasingly pressing challenge (Bowman, 2023).

In the context of this paper, 'steering' refers to the practice of reliably guiding a model's outputs toward or away from displaying the characteristics of a given pattern. A pattern can be specified through humanly interpretable examples, for example demonstrating concepts or behaviors (e.g. love, hate, etc.), but it can also include more complex types of outputs that may not as easily be directly described by humans.

Various methods have been proposed to steer the outputs of LLMs toward desirable or away from (un)desirable patterns, including reinforcement learning from human feedback (RLHF) (Ouyang et al., 2022), supervised fine-tuning (Devlin et al., 2019), and prompt engineering (Liu et al., 2021). RLHF uses human feedback to generate a reward signal, which can then be used in reinforcement learning to steer the models toward the preferred outputs. Supervised fine-tuning adjusts the model's parameters by minimizing the error on a labeled input-output dataset using optimization techniques. Lastly, with prompt engineering, the input prompts are specifically designed to alter the model's outputs without changing the model's parameters itself.

The current methods share common limitations. Both RLHF and supervised fine-tuning require expensive optimization techniques like stochastic gradient descent (SGD) making them computationally expensive (Bottou et al., 2018). Additionally, both methods can fail to generalize due to the variability in human feedback and the specificity of the labeled datasets leading to inconsistent performance (Amodei et al., 2016; Zhang et al., 2023). Prompt engineering, while less computationally expensive, can also have inconsistent steering performance due to the variability and unpredictability of how different prompts affect the model's outputs (Chen et al., 2023).

1.2 Activation Steering

Recently, a new approach has been introduced called *activation steering* (Li et al., 2024; Turner et al., 2024) which aims to predictably control the outputs of LLMs by making direct modifications to the model's activations at inference time. This method benefits from the fact that no changes are made to the model's parameters, eliminating the need for expensive optimization techniques. Additionally, it only requires a small set of prompts instead of a large collection of labeled training data.

Activation steering typically involves caching a set of token activation vectors from an LLM's forward pass on pre-specified prompts. These prompts can demonstrate the desired pattern directly (e.g. functions: "up$\rightarrow$down","happy$\rightarrow$sad", etc.) (Todd et al., 2024) or specify it through contrastive examples (e.g., "love" - "hate") (Turner et al., 2024). In the first case, the cached token activations would be averaged to form the steering vector, in the latter they would be subtracted from each other. This steering vector can then be added or subtracted to the token activations of a new forward pass (regulated by an injection coefficient) at some point (layer) in the processing pipeline to steer the model toward or away from exhibiting the described pattern. This method has shown to be effective at capturing and steering toward or away from a wide range of patterns describing things like specific features/concepts (weddings, love, etc.) (Turner et al., 2024), functions (antonyms, synonyms, etc.) (Todd et al., 2024), and more complex behaviors (truthfulness, power-seeking, etc.) (Panickssery et al., 2024).

This type of steering does have its limitations. Finding suitable contrasting prompts to demonstrate complex patterns can be non-trivial (e.g. for input-output functions). Most importantly, it is not always reliable, sometimes showing inconsistent performance (Turner et al., 2024). This may be explained by the inherent limitations that come with reducing the representation of a complex steering pattern into a single point (vector) in high-dimensional space. Compressing the representation of the pattern this much could lead to a loss of important steering information. Intuitively, it might therefore make more sense to describe these high-dimensional representations in terms of ellipsoidal regions compared to single points.

1.3 Conceptor Steering

This paper introduces an alternative to the current predominant approach for steering LLMs outputs using activation engineering. Instead of averaging or subtracting a set of cached activation vectors to form a steering vector, the cached activations are used to compute a *conceptor*, which we may refer to as a steering matrix. Additionally, instead of manipulating the LLM's activations using vector addition, the activations are (softly) projected using a matrix-vector multiplication with the steering matrix.

Conceptors are mathematical constructs that can be used for the management of neural activations (Jaeger, 2014). A conceptor can be visualized as a structure that describes the activational pattern (state cloud) of a set of high-dimensional activation points using an ellipsoid. This conceptor is mathematically represented by a positive semi-definite matrix with eigenvalues between zero and unity that can be used to (softly) project a new set of activations toward the described ellipsoid. Conceptors have been proven to be effective at controlling pattern-generating recurrent neural networks (RNNs) across a wide range of behaviors (Jaeger, 2017). Furthermore, they have been applied in feed-forward neural networks to prevent catastrophic forgetting and enhance continual learning (He, 2023). Additionally, they have been used for identifying and removing bias subspaces in LLMs such as BERT and GPT (Yifei et al., 2023). Lastly, conceptors have been used to automatically distill linguistic abstractions into a knowledge graph from contextual embeddings (Kuiper, 2024; Bricman, 2022).

Because conceptors are computed from the cloud of activation vectors and encode the correlations between activations (see Section 2.4), conceptors may better capture the activation space of complex patterns compared to simple point representations, which discard information about correlations. Additionally, since conceptors have been successfully applied across various contexts, most importantly in steering pattern-generating RNNs toward a wide range of behaviors (Jaeger, 2017), it is reasonable to also explore their potential steering properties for LLMs. Collectively, these factors provide a strong motivation for investigating how conceptors perform compared to traditional point-based vector methods in reliably steering the output of LLMs.

1.4 Objectives & Methodology

To establish conceptors as a useful steering tool, it is important that they can be used to extract a representation of a pattern from a given set of examples. This representation should then be able to function as a steering mechanism that can be used to modify an LLM's token activations such that its outputs will be steered toward or away from exhibiting the characteristics of the described pattern.

For this paper, we will focus on conceptor steering in the context of input-output functions (e.g. a function that takes a word and returns its antonym). This would be a good first steering goal as it is more complex than simple concepts but simple enough to formalize into a concrete experiment. Furthermore, there is already a rich body of scientific literature on this topic, so we can build on established methodologies and practices (Todd et al., 2024; Jorgensen et al., 2023).

More specifically, we will use the proposed conceptor steering method for extracting and steering the model toward completing specific tasks (input-output functions) and compare its performance against the baseline (no steering) and additive steering. We will also test the methods with and without a potential performance enhancement called *mean-centering.* (Jorgensen et al., 2023) This way, we hope to give a good overview of each method's performance and possible strengths across different contexts.

The effectiveness of the steering mechanisms will be evaluated based on their ability to guide the model towards correctly performing the functions. The performance will be measured using an accuracy metric that compares the model's outputs with the expected (correct) function outputs.

In summary, we aim to answer the question: *will conceptor-based steering methods achieve higher accuracy in steering LLMs toward correctly executing input-output functions compared to traditional point-based steering methods?* Given the potential strengths of conceptors, we hypothesize that *conceptor-based steering methods will achieve higher accuracy in steering LLMs toward correctly executing input-output functions compared to traditional point-based steering methods.*

2 Methods

2.1 The Model

For our experiments, we will make use of a decoder-only transformer neural network. More specifically, we will use *EleutherAI's GPT-J-6B* (Wang & Komatsuzaki, 2021) open-source 6 billion parameter model, pre-trained on the Pile dataset (825GB, ∼300B tokens) (Gao et al., 2020). This was the model of choice by Todd et al. (2024) who showed it to be complex enough to have the representations of various input-output functions (of the kind shown in Section 3.1) encoded in its activation space.

The model contains a stack of 28 transformer layers, each with layer normalization, multi-head attention (MHA), and a feed-forward network (FFN). Central to the architecture is the residual stream, which consists of a sequence of token activation vectors of shape `[num_tokens, d_model]`. Here,

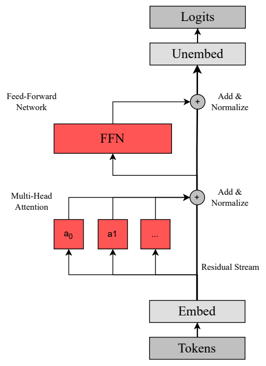


Figure 2.1: High-level diagram of the information processing pipeline of GPT-J-6B, only showing one of 28 transformer layers, inspired by (Elhage et al., 2021).

`num_tokens` equals the number of tokens in the input prompt, and d_model represents the dimensionality of the token embeddings, which for GPT-J-6B is 4096. A diagram of the full structure can be seen in Figure 2.1.

The information processing pipeline begins with the initialization of the residual stream using the embedding(s) of the tokenized input prompt. Each token's embedded activation vector passes through the residual stream. The MHA and FFN components sequentially add information to the residual stream at each transformer layer, with a normalization after each addition. This way, there is a continuous flow and transformation of information throughout the layers of the network. It is for this reason that the residual stream can be conceptualized as the network's "communication channel" (Elhage et al., 2021).

At the end of the forward pass, the final token activation vectors in the residual stream are normalized, unembedded, and processed by a softmax to auto-regressively form the next-token predictions.

2.2 Point-Based Additive Steering

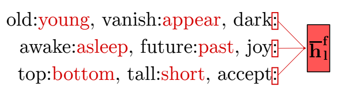
Adding steering vectors to the residual stream has successfully been used to control the output of LLMs across various domains (Turner et al., 2024; Panickssery et al., 2024; van der Weij et al., 2024). The use case that will mainly be focused on here are the findings from the paper by Todd et al. (2024). This paper showed that a steering vector can be extracted from the residual stream that captures the activation space of an input-output function (e.g. a function that takes a word and returns its antonym). This steering vector can then be added to the residual stream at inference time to steer the model toward performing the captured function. For example, by prompting the model with "Hot" and adding the Antonym function vector to the residual stream during a new forward pass, the model would output "Cold".

Their baseline method works as follows. First, a set of in-context learning (ICL) prompts P_f that demonstrate a particular task f (the execution of an input-output function) are compiled. Then for each prompt $p_i^f \in P_f$, the final token activations $h_\ell(p_i^f)$ are cached at a specific layer ℓ from the residual stream h. The cached activation vectors are then averaged into the steering vector $\bar{h}_\ell^f$ for task f at layer ℓ:

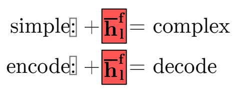
$$\bar{h}_\ell^f = \frac{1}{|P_f|} \sum_{p_i^f \in P_f} h_\ell(p_i^f) \qquad (2.1)$$

To steer the model towards performing this function, the function (steering) vector $\bar{h}_\ell^f$ can be added (without additional re-normalization) to the residual stream at layer ℓ when the model would be completing a prompt containing a previously unseen input. This method has been used to steer the model toward executing a wide range of functions in a zero-shot context meaning that the model was not initially trained to do so, and was also not given explicit examples of this task in its prompt.

This demonstrates that, at particular layers in the model, activation vectors (cached from the residual stream), can encode the execution of a specific function. A steering vector can then be formed


old:young, vanish:appear, dark:
awake:asleep, future:past, joy:
top:bottom, tall:short, accept:
$\bar{\mathbf{h}}_l^f$

(a) Extraction of the antonym function (steering) vector $\bar{h}_\ell^f$ at layer l using 3 ICL prompts.


simple: + $\bar{\mathbf{h}}_l^f$ = complex
encode: + $\bar{\mathbf{h}}_l^f$ = decode

(b) Antonym steering vector in 2 zero-shot contexts.

Figure 2.2: Visualization of how an antonym function (steering) vector can be extracted and applied. Example from (Todd et al., 2024)

.

by taking the average of these cached activation vectors. This steering vector can then be added to the residual stream of a new forward pass to steer the model toward performing the encoded function. See Figure 2.2 for a rough visualization of this method. This paper investigates whether this practice can be improved with the use of Conceptors.

2.3 Mean-Centering for Additive Steering

An important improvement for additive steering is a technique called *mean-centering*, put forward by Jorgensen et al. (2023). This method enhances the effectiveness of steering vectors by reducing the inherent bias present in the activation space of LLMs. Activation vectors in LLMs tend to be anisotropic, meaning that they are not evenly distributed around the origin, but are instead offset in a consistent direction. This can negatively impact the steering vector's performance as the bias vector b representing this offset, does not encode any specific task-related information, diluting the steering vector's effectiveness.

Implementing the mean-centering performance enhancement for steering toward the execution of functions can be done as follows:

First, the steering vector $\bar{h}_\ell^f$ for a specific function f is computed by averaging the activations at layer ℓ on a set of ICL prompts demonstrating the input-output function P_f (as defined in Equation 2.1).

$\bar{h}_\ell^f$ now encodes the task-specific behavior but may still be affected by biases in the model's overall activation space. Mean-centering attempts to mitigate this by subtracting the mean activation of a broader dataset that represents the general activation space of the model. This is done by computing the mean activation vector μ_{train} over a large, representative set of prompts D_{train} from the model's training data (see Appendix B for more details):

$$\mu_{\text{train}} = \frac{1}{|D_{\text{train}}|} \sum_{d \in D_{\text{train}}} h_\ell(d)$$

The mean-centered steering vector can then be obtained by subtracting this general mean activation vector from the task-specific steering vector:

$$\bar{h}_\ell^{f,\text{mc}} = \bar{h}_\ell^f - \mu_{\text{train}} \tag{2.2}$$

This refinement leads to a steering vector that can more effectively guide the model toward the specific task and has been shown to have a positive impact on the overall steering effectiveness (Jorgensen et al., 2023).

2.4 Conceptors

To further enhance the performance of mean-centered point-based additive activation engineering, we introduce a region-based steering mechanism called a conceptor. Conceptors can broadly be defined as a neuro-computational mechanism designed to encapsulate and manipulate the state space of neural activations (Jaeger, 2014).

A conceptor matrix C is a positive semi-definite matrix that captures the principal directions and variances of a set of neural activation vectors. This structure can be visualized as a high-dimensional ellipsoid that describes the overall shape and spread of the activations' "underlying pattern", or state space region. A simple 2D visualization can be seen in Figure 2.3.

One way to formalize the conceptor matrix C, is through an optimization that minimizes the reconstruction error while incorporating a regularization term. The objective function to be minimized is:

$$\min_C \|X - XC\|_F^2 + \alpha^{-2}\|C\|_F^2$$

where X is a matrix of neural activation vectors (stacked as rows), $\|\cdot\|_F$ is the Frobenius norm, and α is the regularization parameter also referred to as the conceptor's aperture. The conceptor matrix that minimizes this objective function is influenced by the aperture parameter α that balances the trade-off between accurately representing the activation pattern and maintaining a generalized representation.

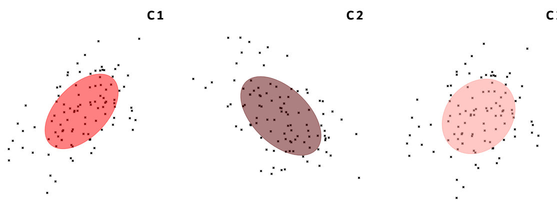


Figure 2.3: 2D visualization of 3 Conceptors that describe the "underlying pattern" or state space region of 3 different sets of neural activations.

The closed-form solution to this problem uses the correlation matrix R of the activation matrix X and the aperture parameter α to compute the conceptor matrix C. The correlation matrix R is defined as:

$$R = \frac{X^T X}{n}$$

where n is the number of samples. The conceptor matrix C can then be computed using the following equation:

$$C(R, \alpha) = R\left(R + \alpha^{-2} I\right)^{-1} \tag{2.3}$$

where I is the identity matrix of the same dimensionality as R.

The eigenvalues μ_i of the conceptor matrix C are defined as:

$$\mu_i = \begin{cases} \frac{\lambda_i}{\lambda_i + \alpha^{-2}} & \text{for } 0 < \lambda_i < 1 \text{ and } 0 < \alpha < \infty \\ 0 & \text{for } 0 < \lambda_i < 1 \text{ and } \alpha = 0 \\ 1 & \text{for } 0 < \lambda_i < 1 \text{ and } \alpha = \infty \\ 0 & \text{for } \lambda_i = 0 \text{ and } 0 \leq \alpha \leq \infty \\ 1 & \text{for } \lambda_i = 1 \text{ and } 0 \leq \alpha \leq \infty \end{cases}$$

where λ_i represents the eigenvalues of the correlation matrix R. These eigenvalues μ_i fall within the interval $[0, 1]$ and are influenced by the aperture parameter α. When α is large, the eigenvalues μ_i approach 1 and C approaches the identity matrix, causing the conceptor to allow for more signal components to pass through the projection of the states with the conceptor matrix Cx. Conversely, when α is small, the eigenvalues μ_i approach 0, causing the conceptor to allow for less variability. In the extreme case of $\alpha = 0$, the conceptor collapses to the zero mapping.

Using this mechanism, the conceptor matrix C can be used to steer a new activation vector $\mathbf{x}$ with a matrix-vector multiplication:

$$\mathbf{x}' = C\mathbf{x}$$

where $\mathbf{x}'$ is the conceptor-steered activation vector. We can think of this operation as a "soft projection". A projection matrix has eigenvalues that are either zero or unity, but the conceptor matrix has "soft" values that can lie between zero and unity. Thus, the operation "softly projects" the activation vector $\mathbf{x}$ toward the pattern represented by C by scaling its components according to the patterns' principal directions. As a result, the directions and magnitudes of the activation vector $\mathbf{x}$ are adjusted to align more closely with the desired pattern captured by C.

3 Experimental Setup

The experimental setup aims to measure and compare the conceptor and point-based additive steering mechanisms (with and without mean-centering) on their ability to steer the model towards correctly executing a set of functions. For each function, the described experiment will be repeated 5 times with different random seeds. For each experiment, a new set of steering mechanisms will be generated (additive, additive + mean-centering, conceptor, conceptor + mean-centering). These mechanisms will be used to steer the model toward executing the described function on independent forward passes. For each of the five experiments, the steering performances of each mechanism will be measured and compared.

3.1 Data

The examples of the input-output functions come from the dataset used in the experiment by Todd et al. (2024)*. We will refer to this as the function pairs dataset. For a range of functions, it provides a large set of (input, output) pairs in JSON format. The specific functions used for our experiments are the same ones covered in the mean-centering experiment by Jorgensen et al. (2023), namely: antonyms (e.g. good→bad), present-past (e.g. go→went), English-French (e.g. hello→bonjour), singular-plural (e.g. mouse→mice), country-capital (e.g. Netherlands→Amsterdam), and capitalize (e.g. word→Word).

3.2 Hyperparameters & Configurations

The experimental outcomes are influenced by several carefully fine-tuned hyperparameters that aim to maximize the performance of all steering mechanisms. The hyperparameters that were fine-tuned include the aperture α (α_{reg} for regular and α_{mc} for mean-centered conceptors), an injection coefficient β_{add} for additive steering, and a rescaling

*Link to the function pairs dataset

coefficient β_c for conceptor steering. Each hyperparameter was optimized through a grid search, the details of which can be found in Appendix A.

All the experimental configurations (number of experiments, number of ICL prompts and examples per prompt, accuracy metric, etc.) were unless mentioned otherwise, adopted from the paper by Todd et al. (2024) to ensure comparability of results.

3.3 Steering Mechanism Generation

For each experiment, to generate the 4 steering mechanisms, we first compile $N_p = 100$ (ICL) prompts that demonstrate the respective input-output function. The prompts are formed by randomly sampling $N = 10$ input-output pairs from the function pairs dataset. If for a specific function, the dataset contains less than $N_p \times N = 1000$ input-output examples, this sampling is done with replacement. For each prompt p_i^f, the last input-output pair has the output stripped, resulting in the format:

$$p_i^f = "x_1 : y_1, x_2 : y_2, ..., x_{N-1} : y_{N-1}, x_N : "$$

where x represents the input tokens of a randomly sampled (input, output) pair, y represents the corresponding output tokens, N represents the number of sampled input-output pairs, and $i \in \{1, \ldots, N_p\}$. A very simple example where $N_p = 3$ and $N = 3$ can be seen in Figure 2.2a.

Formally, for each function $f \in F$ in our set of in-context learning (ICL) tasks, we have compiled a set P_f of ICL prompts $p_i^f \in P_f$. Each prompt p_i^f is a sequence of tokens with N input-output exemplar pairs (x, y) that demonstrate the function f mapping between x and y. For each experiment, we generate N_p such prompts.

Now that the ICL prompts have been generated, we need to extract the relevant activations. Todd et al. (2024) showed that the neural representations of the functions are encoded in the activation vector of the last token (":") of the prompt, right before the transformer would auto-regressively start generating the output token(s). Moreover, the point in the residual stream h at which the functions were most strongly encoded was shown to be at the beginning of layers $L = \{9, \ldots, 16\}$, right before MHA and FFN (Todd et al., 2024).

Formally, for each function $f \in F$ and each prompt $p_i^f \in P_f$, the activation vectors $h_\ell^f(p_i^f)$ are extracted from the residual stream h at each relevant layer $l \in L$ from the last token's (":") activation vector.

For each function $f \in F$ and each layer $l \in L$, we now have N_p cached activation vectors $h_\ell^f(p_i^f)$ aimed to encode the neural representation of f at layer l. Using this, we can generate the layer-specific steering mechanisms for each function as follows:

- The standard additive steering mechanism $\bar{h}_\ell^f$ is generated by averaging over all the cached activation vectors $h_\ell^f(p_i^f)$ respectively as described in Equation 2.1.
- The additive steering mechanism with mean-centering $\bar{h}_\ell^{f,\mathrm{mc}}$ is computed by taking the previously generated steering mechanism $\bar{h}_\ell^f$ and subtracting μ_{train} as described in Equation 2.2.
- The regular conceptor steering mechanism C is computed as described in Equation 2.3 using the aperture value α_{reg}. The correlation matrix R is computed as $R = \frac{X^T X}{N_p}$, where X is the matrix of all $h_\ell^f(p_i^f)$ stacked activation vectors.
- The mean-centered conceptor steering mechanism C^{mc} is computed with some minor adjustments. The matrix X is formed by subtracting μ_{train} from the activation vectors $h_\ell^f(p_i^f)$ before stacking them. This results in an adjusted correlation matrix R:

  $$R = \frac{(X - \mu_{\mathrm{train}})^T (X - \mu_{\mathrm{train}})}{N_p}$$

  The mean-centered conceptor matrix C^{mc} can then be calculated as described in Equation 2.3 using the aperture value α_{mc} and the adjusted correlation matrix R.

3.4 Experimental Procedure

To test the performance of the generated steering mechanisms, new sets of $N_t = 1000$ input-output pairs are randomly sampled from the function pairs dataset for each experiment. This is done with replacement for functions where the dataset contains less than N_t pairs. An input prompt p_t is formatted as $p_t = "x : "$, where x is a tokenized input from an input-output pair. The tokenized output y from the pair is left out from p_t as it will be used to test the accuracy of the steering mechanisms. For each experiment, we now have N_t test input prompts p_t.

To test the accuracy of the steering mechanisms, we apply the layer-specific steering mechanisms on independent forward passes and record their subsequent output. This means that for our experimental configuration, across the functions $f \in F$, the 5 experiments, the 4 steering mechanisms (excluding the baseline), the N_t number of test prompts, and the number of layers $l \in L$, there will be $6 \times 5 \times 4 \times 1000 \times 8 = 960,000$ forward passes, each with a steering intervention.

Each steering intervention will consist of a layer-specific steering mechanism modifying the residual stream h at the mechanisms' respective layer l. This modification can be defined as transforming the unmodified residual stream activation vector h_ℓ into the steered activation vector h'_ℓ. The steering mechanisms' modification can be described as follows:

- For the standard additive steering mechanism, the averaged activation vector $\bar{h}^f_\ell$ is multiplied by the injection coefficient β_{add} and added to the residual stream activation vector h_ℓ:
$$h'_\ell = \beta_{\text{add}}\,\bar{h}^f_\ell + h_\ell$$
- For the additive steering mechanism with mean-centering, the mean-centered average activation vector $\bar{h}^{f,\text{mc}}_\ell$ is multiplied by the injection coefficient β_{add} and added to the residual stream activation vector h_ℓ:
$$h'_\ell = \beta_{\text{add}}\,\bar{h}^{f,\text{mc}}_\ell + h_\ell$$
- For the regular conceptor steering mechanism, the residual stream activation vector h_ℓ is multiplied using the conceptor matrix C and further multiplied with the rescaling coefficient β_c:
$$h'_\ell = \beta_\text{c}\,C\,h_\ell$$
- For the mean-centered conceptor steering mechanism, the residual stream activation vector h_ℓ is first adjusted by subtracting μ_{train}. This adjusted vector is then multiplied with the mean-centered conceptor matrix C^{mc} and further multiplied with the rescaling coefficient β_c. Finally, μ_{train} is added back to the result:
$$h'_\ell = \beta_\text{c}\,C^{\text{mc}}\,(h_\ell - \mu_{\text{train}}) + \mu_{\text{train}}$$
- For the baseline condition, no modifications are made to the residual stream.
$$h'_\ell = h_\ell$$

After the respective modifications have been made to the residual stream, the forward passes will continue as usual. At the end of each forward pass, the final logits are converted into probabilities using a softmax, and the token with the highest probability is selected. This means that at the end of one experiment, we have N_t single-token outputs for each layer-specific steering mechanism. These tokens can now be compared with the first token of output y that corresponds with the input x of the initial prompt p_t. Based on how many of the N_t outputs were correctly identified, a top-1 accuracy is calculated for each layer-specific steering mechanism. This experiment is repeated 5 times for each function $f \in F$ to account for variability caused by the random sampling for the generation of the steering mechanisms and test sets.

4 Results

This section provides a comparison of the four steering mechanisms (additive, additive with mean-centering, conceptor, and conceptor with mean-centering) across six functions: antonym, capitalization, country-capital, English-French, present-past, and singular-plural. The performance is evaluated based on the top-1 accuracy of the layer-specific steering mechanisms that make interventions on layers 9 to 16. The baseline performance is consistently low across all tasks, so it is not included in the individual function discussions.

4.1 Function Steering Results

The following subsection outlines the performance of each layer-specific steering mechanism on how well it was able to steer the model toward correctly executing each of the six functions.

- **Antonyms**: both additive methods show moderate improvements, with the mean-centered mechanisms performing slightly better across all layers. Conceptor methods, particularly with mean-centering, achieve a significantly higher accuracy, peaking around layers 12 and 13.

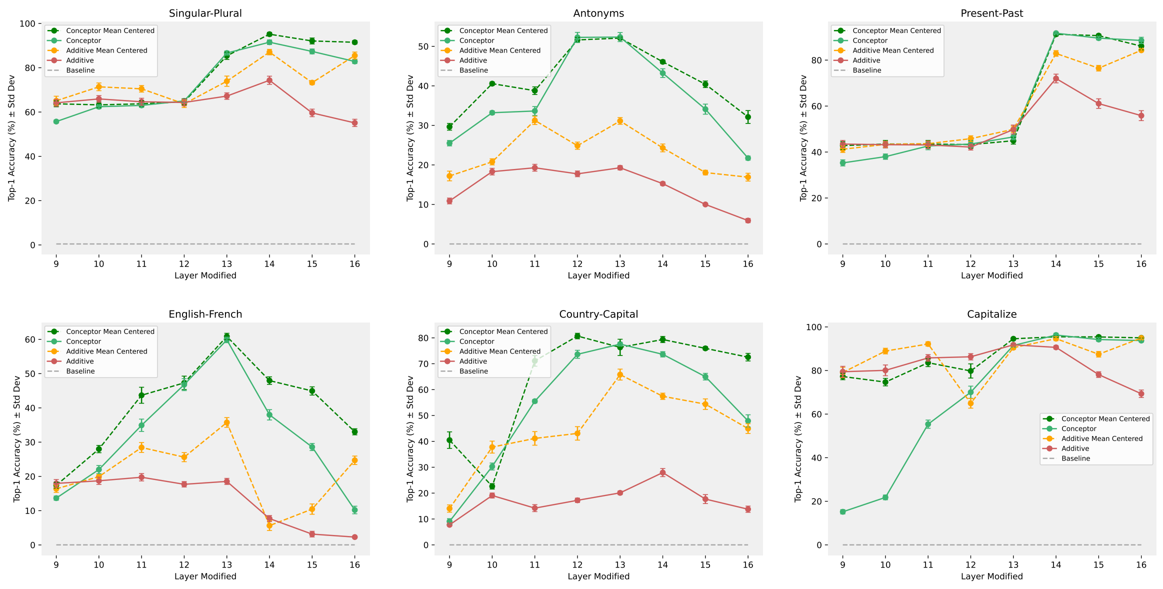


Figure 4.1: Plots displaying the layer-specific performance of the 4 steering mechanisms for 6 different functions.

- **Capitalization**: all four methods (except for the regular conceptor method in the beginning layers) show significant improvements, reaching near-perfect performance.
- **Country-Capital**: additive methods show moderate improvements, with additive mean-centered performing significantly better. Conceptor methods, especially when mean-centered, achieve the highest performance, peaking around layers 12 and 14.
- **English-French**: Although both additive methods show slight improvements, the conceptor methods significantly outperform them, with the mean-centered mechanism showing the highest accuracy across all layers.
- **Present-Past**: all four methods show equal moderate improvements in the early layers. Around layer 14, the performance of the methods increases significantly, with the conceptor methods outperforming the additive methods.
- **Singular-Plural**: Although both additive methods show consistent improvements, the conceptor methods outperform additive them, with mean-centered conceptors showing more gains, especially in layers 13 to 15.

4.2 Overall Task Performance

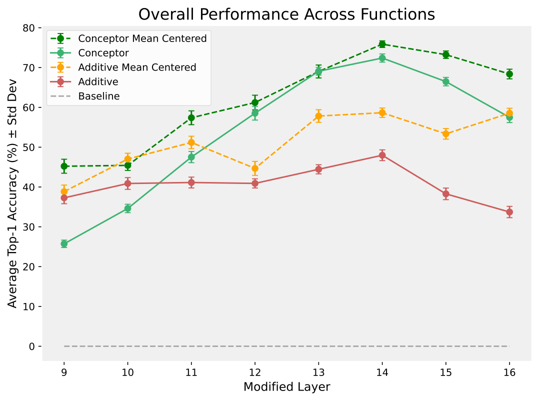


Figure 4.2: Averaged performance of the four steering mechanisms across the 6 tested functions.

The overall performance across all tasks and layers is outlined in Figure 4.2. This plot averages the top-1 accuracy across all functions for each layer and steering mechanism.

It can be seen that in the early layers where the overall performance is low, conceptors slightly underperform compared to the additive methods. But from layer 11 onward, the conceptor methods outperform the additive methods, peaking around layer 14.

4.3 Performance at Best-performing Layers

The steering performances at layers 14 and 15 across all tasks were also analyzed in more detail. These layers were chosen as across all steering mechanisms, the performance was highest at these locations indicating that this is where the functions' representations are most strongly encoded in the model. This is outlined in Figure 4.3 where it can be observed that both conceptor methods significantly outperformed the additive methods on all functions.

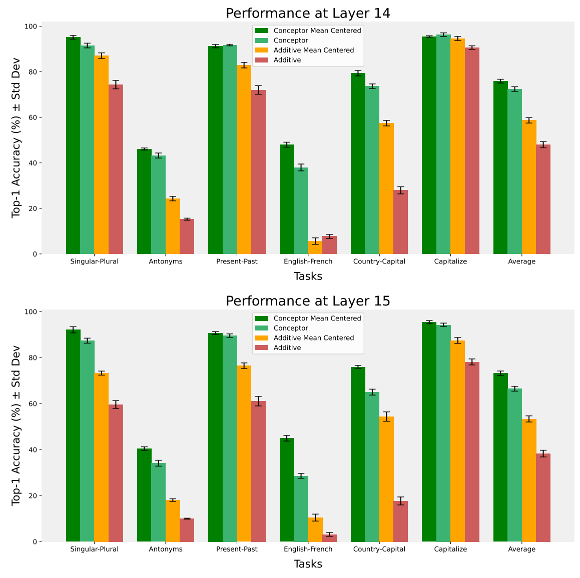


Figure 4.3: 2 plots showing the steering performance of the 4 steering mechanisms on all 6 tested functions for the best-performing layers.

4.4 Statistical Significance

To validate the observed differences between the steering mechanisms, statistical significance tests were conducted. These tests help determine if the improvements seen with mean-centered conceptor and mean-centered additive approaches are statistically significant. We used independent t-tests to compare the performance at layer 14 for each task and a repeated measures ANOVA test to assess the overall performance across all layers and tasks. A significance level of 0.05 was used for all tests.

4.4.1 T-test

An independent one-tailed t-test was performed to compare the performance of the mean-centered conceptor and mean-centered additive methods at layer 14 for each task. We focused on layer 14 because it consistently showed the highest average performance across all tasks and steering mechanisms, suggesting that this layer is where the functional patterns are most richly encoded. We chose to only focus on comparing the mechanisms with mean-centering as these generally performed better across tasks. The null hypothesis was that the performance of the mean-centered conceptor method is not significantly greater than the performance of the mean-centered additive methods.

- **Antonyms:** The t-test between mean-centered conceptor (M = 46.08, SD = 0.82) and mean-centered additive steering (M = 24.3, SD = 0.99) at layer 14 showed a significant difference, t(8) = 37.91, p = 1.287e-10.

- **Capitalize:** The t-test between mean-centered conceptor (M = 95.48, SD = 0.32) and mean-centered additive steering (M = 94.62, SD = 0.92) at layer 14 showed a significant difference, t(8) = 1.98, p = 0.042.

- **Country-Capital:** The t-test between mean-centered conceptor (M = 79.36, SD = 1.19) and mean-centered additive steering (M = 57.42, SD = 1.21) at layer 14 showed a significant difference, t(8) = 28.95, p = 1.097e-09.

- **English-French:** The t-test between mean-centered conceptor (M = 47.96, SD = 1.11) and mean-centered additive steering (M = 5.64, SD = 1.43) at layer 14 showed a significant difference, t(8) = 52.37, p = 9.791e-12.

- **Present-Past:** The t-test between mean-centered conceptor (M = 91.24, SD = 0.75) and mean-centered additive steering (M = 82.9, SD = 1.23) at layer 14 showed a significant difference, t(8) = 12.95, p = 5.993e-07.

- **Singular-Plural:** The t-test between mean-centered conceptor (M = 95.14, SD = 0.82) and mean-centered additive steering (M = 87.06, SD = 1.24) at layer 14 showed a significant difference, t(8) = 12.17, p = 9.642e-07.
- **Overall performance across all tasks:** The t-test between mean-centered conceptor (M = 75.88, SD = 27.47) and mean-centered additive steering (M = 58.99, SD = 33.66) at layer 14 showed a significant difference, t(58) = 64.04, p = 8.082e-56.

The results indicate that the null hypothesis was rejected in all instances, showing that the mean-centered conceptor method significantly outperformed the mean-centered additive method for all functions at layer 14.

4.4.2 Repeated Measures ANOVA

A repeated measures ANOVA test was conducted to compare the performance of the mean-centered additive and mean-centered conceptor methods across all layers and tasks. This analysis helps determine which steering mechanism performs better overall while accounting for the repeated measurements within tasks.

- **Main effect of layer:** There was a significant main effect of layer on performance, F(7, 35) = 4.519, p = 0.0011.
- **Main effect of steering mechanism:** There was a significant main effect of the steering mechanism, F(1, 5) = 7.625, p = 0.0398, with mean-centered conceptor steering (M = 68.27, SD = 23.69) outperforming mean-centered additive steering (M = 52.39, SD = 28.45) overall.
- **Interaction effect:** The interaction between the layer and steering mechanism was significant, F(7, 35) = 3.726, p = 0.0041. This indicates that while mean-centered conceptor steering generally outperformed mean-centered additive steering, the degree of this difference varied across layers.

4.5 Summary

The conceptor steering mechanisms, particularly with mean-centering, had superior performance across all tasks and layers compared to the additive methods. The statistical significance tests, including independent t-tests for performance at layer 14, and a repeated measures ANOVA for overall performance across all layers and tasks, confirmed that these improvements are statistically significant. The results suggest that conceptor methods are more effective at capturing and steering LLMs toward correctly executing functions. Further implications of these findings will be analyzed in the discussion and conclusion sections.

5 Discussion

5.1 Interpretation of Results

The results align with our initial hypothesis that conceptor methods result in a higher steering performance than point-based methods. Our findings showed that, across several contexts, conceptor-based steering mechanisms performed better compared to point-based steering mechanisms. This suggests that conceptors may offer a more precise and flexible approach to capturing the representation of patterns in the activation spaces of LLMs. In addition to this, we found that the mean-centering enhancement appears to boost the performance even further. This indicates that just like with point-based methods, reducing the effects of anisotropy in LLMs can improve the steering effectiveness of conceptors.

In addition to this, the conceptor methods consistently performed best in layers where the patterns were thought to be most strongly encoded. However, in layers with lower overall performance, the differences between the mechanisms were less consistent. This indicates that conceptor methods excel when applied to layers where the pattern is likely to be encoded, but their advantage is less clear in other layers.

5.2 Comparison with Previous Work

Todd et al. (2024) showed that function (steering) vectors can be extracted and injected in the forward pass of an LLM to steer it toward executing a set of input-output functions. We successfully replicated their findings using the same experimental configurations. In addition to this, we were able to build on their research, showing that the steering mechanism does not just have to be limited to a point-based (vector) representation, but in our tested contexts, can even benefit from a region-based representation using conceptors.

Moreover, Jorgensen et al. (2023) showed that the inherent bias (anisotropy) in the activation space of LLMs can reduce the effectiveness of the extracted point-based (function) steering mechanisms. By removing this bias from the extracted mechanism, its steering performance was shown to be improved. We successfully replicated these findings and showed that this method (mean-centering) can also improve the steering performance of region-based conceptor steering mechanisms.

Lastly, several studies showed that conceptors are a useful neuro-computational mechanism for the management of neural activations in several contexts. This includes controlling pattern-generating RNNs (Jaeger, 2017), enhancing continual learning in FFNs (He, 2023), reducing bias in LLMs (Yifei et al., 2023), and creating knowledge graphs from contextual embedding spaces (Kuiper, 2024; Bricman, 2022). Our study supports that just like in the aforementioned studies, conceptors can be used for many types of neural management applications, in this case highlighting their usability in steering toward patterns in the activation space of LLMs.

5.3 Limitations

5.3.1 Methodological Limitations

Although we performed some fine-tuning to identify the optimal parameters for the injection/rescaling coefficients (β_{add} and β_{c}) and the apertures (α_{reg} and α_{mc}), these parameters were computed only based on performances on the antonym task. It is conceivable that specific layers or tasks might benefit from slight parameter modifications, potentially leading to different outcomes. Unfortunately, our ability to perform more extensive fine-tuning was limited by computational restraints.

Our study focused solely on steering LLMs toward executing functions, and just like in the experiments by Todd et al. (2024), only the first output token was checked for correctness. Consequently, it is not immediately clear what the implications of this steering method are on grammar or overall model capabilities. More complex experiments, including testing for more complex behaviors, would need to be conducted to examine this.

The experiments by Todd et al. (2024) demonstrated that the effectiveness of function (steering) vectors scales to a set of larger models that they experimented with, including *GPT-NeoX* and *Llama 2 (70B)*. While there is no reason to believe that for conceptor steering, this would not scale similarly, it has not been proven in our experiment.

Manual inspection of the function pairs dataset revealed that some of the outputs were quite ambiguous and non-trivial (e.g. for antonym, 'cavity$\rightarrow$healthy tooth'), potentially limiting the performance potential. Especially given that the accuracy metric used was top-1 accuracy, it may not have given the model enough flexibility to perform optimally.

5.3.2 Conceptor Challenges

Using conceptors to steer LLMs does present its own set of challenges. To form a meaningful conceptor that accurately captures an underlying pattern from a set of neural activations, more than one data point is required, and preferably many more. This is in contrast to point-based steering mechanisms where technically only a single activation point is needed. That said, in practice, often many points are averaged to more accurately describe the desired pattern using a point-based representation (Todd et al., 2024).

Generating a conceptor is relatively more expensive compared to generating a point-based steering mechanism due to the necessary extra computational steps. This includes computing a correlation matrix and performing a matrix inversion, instead of simply averaging over a set of vectors. Additionally, conceptors also take up more space in memory as they are represented as a matrix instead of a vector. Lastly, applying a conceptor is also slightly more computationally expensive as it involves a matrix-vector multiplication instead of a simple vector addition.

The aperture parameter, which acts as a regularization term, needs careful calibration. Proper fine-tuning is needed to ensure that the ellipsoidal representation of the conceptor strikes a good balance between maintaining a generalized representation and precisely capturing the principal directions and variances of the desired pattern.

Despite their potential, much more research is needed to fully establish the best practices and limitations of the use of conceptors in the context of LLMs. This includes understanding the most effective ways to use conceptors across different tasks and models, as well as identifying any limitations that may restrict their broader applicability.

5.4 Wider Implications

This paper suggests a fundamental shift in how activation steering is typically performed. It hints towards a paradigm shift in how we think about capturing and steering toward patterns in neural activation spaces. More specifically, suggesting that region-based representations may allow for more flexible and nuanced steering compared to point-based representations. This paper demonstrates that this is not only possible but can actually increase the overall steering performance. Although this research primarily addresses simple input-output functions, it lays the groundwork for investigating more complex behaviors. If more research is done and conceptor-based steering continues to be demonstrated as a useful method for steering LLMs, it could have significant positive implications for debiasing models, aligning models with human values, and overall AI safety.

Furthermore, given that the proposed steering method is less expensive than some of the current methods (RLHF and supervised fine-tuning) that require expensive optimization techniques (Bottou et al., 2018), it has a low alignment tax (i.e. the additional cost of opting for safety) (Turner et al., 2024). This economic advantage increases the feasibility of its widespread adoption, making it relevant for both current and future frontier AI models.

5.5 Future Research

There are several areas that future research on conceptor steering could focus on to address the limitations and maximize their utility.

Firstly, it is crucial to investigate the effect of different parameters such as the injection/rescaling coefficients (β_{add} and β_{c}) and the apertures (α_{reg} and α_{mc}), on the steering mechanisms' performance across different contexts. Establishing standards for determining the right parameters and how to most efficiently find them would make conceptors much simpler and cheaper to work with.

Another promising extension of this research would be multi-layer steering. Given that conceptors have been successful in steering the behavior of pattern-generating RNNs which inherently use dynamic time-series activation data, they might also be effective in capturing and steering patterns across multiple LLM layers. Exploring this potential could lead to even more nuanced control and possibly further enhance the steering performance.

Additionally, although combining steering vectors could potentially allow for a more nuanced way to steer toward more complex patterns, it has not been shown to work reliably yet with point-based representations (van der Weij et al., 2024). This could potentially be solved using conceptors that support boolean operations (like NOT, AND, OR) (Jaeger, 2017). The ability to perform logical operations on conceptors could be very useful for steering a model toward or away from more complex patterns that a single conceptor may not as easily capture. We conducted a simple experiment where two conceptors representing different functions (singular-plural and capitalize) were combined using the OR operator. This new conceptor was successfully able to steer an LLM towards executing a new 'singular-capitalized plural' function. Even more promising is that its steering accuracy was almost twice as high compared to the point-based steering mechanism formed by adding the two individual mechanisms to each other. The results can be found in Appendix D.

To establish conceptors as versatile and universal mechanisms for steering LLMs, it is important to test their steering effectiveness on a broader set of tasks and behaviors. This includes an evaluation of the impact on the model's grammatical and overall capabilities, ensuring that these are not negatively affected. One way this could be achieved is by building on the work by Turner et al. (2024) and Panickssery et al. (2024) who did this for point-based steering methods. They used point-based (contrastive additive) steering mechanisms to steer different LLMs toward a broad set of concepts and behaviors, and tested the overall steering impact on the models' capabilities. As a proof-of-concept, we modified the (open-sourced) experiments from Turner et al. (2024) to work with conceptors in addition to point-based steering methods. Preliminary results outlined in Appendix C show that conceptors can successfully be used to steer an LLM, more specifically *GPT-2-XL* (Radford et al., 2019), toward the concepts of ”love” and ”wedding”, producing multi-token syntactically sensible outputs.

Finally, an important detail to look into is scalability. While this study showed that conceptors can be used as an effective steering mechanism for the GPT-J-6B model, it is important to prove that this scales to larger models as well. This way it will remain useful as LLMs continue to scale in size. Future work could explore whether conceptor-based steering methods naturally scale to larger models and examine the possible associated increase in computational expenses. Moreover, if other NLP architectures become competitive to transformers (e.g., state space models), it would be valuable to investigate whether conceptor-based steering methods transfer to those architectures as well.

6 Conclusion

This study introduced conceptors as a novel mechanism for steering large language models through activation engineering. We theorized that the region-based structure of conceptors would provide more nuanced and precise steering control over the activation space of LLMs compared to traditional point-based representations. The results of our experiments supported this as conceptor-based steering mechanisms outperformed the traditional point-based steering methods across all tested tasks. These findings suggest that conceptors may be a promising tool for controlling the outputs of LLMs. However, to establish them as a universally versatile steering mechanism, further research is needed to better understand their limitations and applicability across broader contexts.

References

Amodei, D., Olah, C., Steinhardt, J., Christiano, P., Schulman, J., & Mané, D. (2016). *Concrete problems in ai safety.* Retrieved from https://arxiv.org/abs/1606.06565

Bottou, L., Curtis, F. E., & Nocedal, J. (2018). *Optimization methods for large-scale machine learning.* Retrieved from https://arxiv.org/abs/1606.04838

Bowman, S. (2023). Eight things to know about large language models. *ArXiv*, *abs/2304.00612*. doi: 10.48550/arXiv.2304.00612

Bricman, P. (2022, July). *Nested state clouds: Distilling knowledge graphs from contextual embeddings.* Retrieved from https://fse.studenttheses.ub.rug.nl/27840/ (Bachelor's Project Thesis, University of Groningen, Supervisors: Prof. Dr. Herbert Jaeger, Dr. Jacolien van Rij-Tange)

Bubeck, S., Chandrasekaran, V., Eldan, R., Gehrke, J., Horvitz, E., Kamar, E., ... Zhang, Y. (2023). Sparks of artificial general intelligence: Early experiments with GPT-4. *ArXiv*, *abs/2303.12712*. doi: 10.48550/arXiv.2303.12712

Chang, Y.-C., Wang, X., Wang, J., Wu, Y., Zhu, K., Chen, H., ... Xie, X. (2023). A survey on evaluation of large language models. *ArXiv*, *abs/2307.03109*. doi: 10.48550/arXiv.2307.03109

Chen, B., Zhang, Z., Langren'e, N., & Zhu, S. (2023). Unleashing the potential of prompt engineering in large language models: a comprehensive review. *ArXiv*, *abs/2310.14735*. doi: 10.48550/arXiv.2310.14735

Devlin, J., Chang, M.-W., Lee, K., & Toutanova, K. (2019). *Bert: Pre-training of deep bidirectional transformers for language understanding.* Retrieved from https://arxiv.org/abs/1810.04805

Elhage, N., Nanda, N., Olsson, C., Henighan, T., Joseph, N., Mann, B., ... Olah, C. (2021). A mathematical framework for transformer circuits. *Transformer Circuits Thread*. (https://transformer-circuits.pub/2021/framework/index.html)

Gallegos, I. O., Rossi, R. A., Barrow, J., Tanjim, M. M., Kim, S., Dernoncourt, F., ... Ahmed, N. K. (2024). *Bias and fairness in large language models: A survey.* Retrieved from https://arxiv.org/abs/2309.00770

Gao, L., Biderman, S., Black, S., Golding, L., Hoppe, T., Foster, C., … others (2020). The pile: An 800gb dataset of diverse text for language modeling. *arXiv preprint arXiv:2101.00027*.

Gokaslan, A., Cohen, V., Pavlick, E., & Tellex, S. (2019). *Openwebtext corpus.* http://Skylion007.github.io/OpenWebTextCorpus.

He, O. (2023). *Continual lifelong learning in neural systems: overcoming catastrophic forgetting and transferring knowledge for future learning* (Doctoral dissertation, University of Groningen). doi: 10.33612/diss.625549871

Jaeger, H. (2014). *Conceptors: an easy introduction.* Retrieved from https://arxiv.org/abs/1406.2671

Jaeger, H. (2017). *Controlling recurrent neural networks by conceptors.* Retrieved from https://arxiv.org/abs/1403.3369

Jorgensen, O., Cope, D., Schoots, N., & Shanahan, M. (2023). *Improving activation steering in language models with mean-centring.* Retrieved from https://arxiv.org/abs/2312.03813

Kuiper, J. (2024). *Using conceptors to extract abstraction hierarchies from corpora of natural text: Combatting word polysemy using word sense disambiguation techniques* (Master's Thesis / Essay). University of Groningen, Groningen, Netherlands. (Date Deposited: 23 Jan 2024 14:46, Last Modified: 23 Jan 2024 14:46)

Li, K., Patel, O., Viégas, F., Pfister, H., & Wattenberg, M. (2024). *Inference-time intervention: Eliciting truthful answers from a language model.* Retrieved from https://arxiv.org/abs/2306.03341

Liu, P., Yuan, W., Fu, J., Jiang, Z., Hayashi, H., & Neubig, G. (2021). *Pre-train, prompt, and predict: A systematic survey of prompting methods in natural language processing.* Retrieved from https://arxiv.org/abs/2107.13586

Ouyang, L., Wu, J., Jiang, X., Almeida, D., Wainwright, C. L., Mishkin, P., … Lowe, R. J. (2022). Training language models to follow instructions with human feedback. *ArXiv*, *abs/2203.02155*. doi: 10.48550/arXiv.2203.02155

Pan, Y., Pan, L., Chen, W., Nakov, P., Kan, M.-Y., & Wang, W. Y. (2023). *On the risk of misinformation pollution with large language models.* Retrieved from https://arxiv.org/abs/2305.13661

Panickssery, N., Gabrieli, N., Schulz, J., Tong, M., Hubinger, E., & Turner, A. M. (2024). *Steering llama 2 via contrastive activation addition.* Retrieved from https://arxiv.org/abs/2312.06681

Radford, A., Wu, J., Child, R., Luan, D., Amodei, D., Sutskever, I., et al. (2019). Language models are unsupervised multitask learners. *OpenAI blog*, *1*(8), 9.

Shevlane, T., Farquhar, S., Garfinkel, B., Phuong, M., Whittlestone, J., Leung, J., … Dafoe, A. (2023). *Model evaluation for extreme risks.* Retrieved from https://arxiv.org/abs/2305.15324

Todd, E., Li, M. L., Sharma, A. S., Mueller, A., Wallace, B. C., & Bau, D. (2024). *Function vectors in large language models.* Retrieved from https://arxiv.org/abs/2310.15213

Turner, A. M., Thiergart, L., Leech, G., Udell, D., Vazquez, J. J., Mini, U., & MacDiarmid, M. (2024). *Activation addition: Steering language models without optimization.* Retrieved from https://arxiv.org/abs/2308.10248

van der Weij, T., Poesio, M., & Schoots, N. (2024). *Extending activation steering to broad skills and multiple behaviours.* Retrieved from https://arxiv.org/abs/2403.05767

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., … Polosukhin, I. (2023). *Attention is all you need.* Retrieved from https://arxiv.org/abs/1706.03762

Wang, B., & Komatsuzaki, A. (2021, May). *GPT-J-6B: A 6 Billion Parameter Autoregressive Language Model.* https://github.com/kingoflolz/mesh-transformer-jax.

Xu, B., & Poo, M. (2023). Large language models and brain-inspired general intelligence. *National Science Review*, *10*. doi: 10.1093/nsr/nwad267

Yifei, L. S., Ungar, L., & Sedoc, J. (2023). *Conceptor-aided debiasing of large language models.* Retrieved from `https://arxiv.org/abs/2211.11087`

Zhang, K., Wen, Q., Zhang, C., Cai, R., Jin, M., Liu, Y., … Pan, S. (2023). Self-supervised learning for time series analysis: Taxonomy, progress, and prospects. *ArXiv*, *abs/2306.10125*. doi: 10.48550/arXiv.2306.10125

Zhao, W. X., Zhou, K., Li, J., Tang, T., Wang, X., Hou, Y., … rong Wen, J. (2023). A survey of large language models. *ArXiv*, *abs/2303.18223*. doi: 10.48550/arXiv.2303.18223

A Hyperparameter Optimization

The performance of the steering mechanisms in the experiments was optimized through a grid search over four key hyperparameters (based on just the antonym function steering accuracy): the aperture (α), the injection coefficient (β_{add}), and the rescaling coefficient (β_{c}).

For the aperture parameters, both the regular aperture (α_{reg}) and the mean-centered aperture (α_{mc}) were optimized by initially searching over a wide range of values $\{10^{-4}, 10^{-3}, \dots, 10^{3}, 10^{4}\}$. Upon identifying promising regions, the search was refined with intervals relative to the identified scales, achieving a precision of 0.0125. The final optimized values were $\alpha_{\text{reg}} = 0.0125$ and $\alpha_{\text{mc}} = 0.05$.

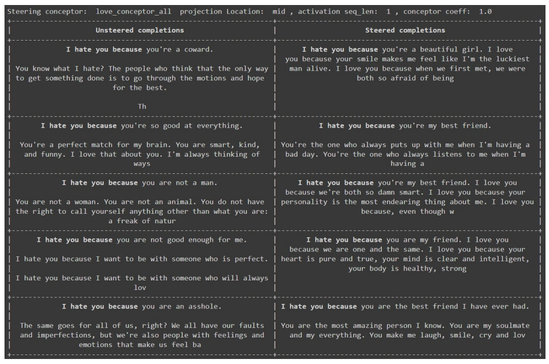
The injection coefficient (β_{add}) and the rescaling coefficient (β_{c}) were optimized in a two-stage process. An initial search was conducted over the range [0, 10] with an interval of 1. After identifying a suitable range, a finer search was performed within a ± 1 window around the best initial estimate, using an interval of 0.1. This process led to the final values of $\beta_{\text{add}} = 2.3$ and $\beta_{\text{c}} = 3.9$.

B Mean Training Data

The mean activation vector μ_{train} was calculated using the same procedure described by Jorgensen et al. (2023): A subset from the dataset used to train GPT-2 was compiled (Gokaslan et al., 2019). The subset was constructed by storing all entries from the folders *urlsf subset01-1 data* and *urlsf subset01-182 data*. After this, only entries that contained less than 500 tokens (using the GPT-2 Tokenizer) were retained. This resulted in 210 entries from which the final 10 were removed, leaving a dataset of 200 entries. The mean activation vector μ_{train} was then computed by averaging the activations over this dataset.

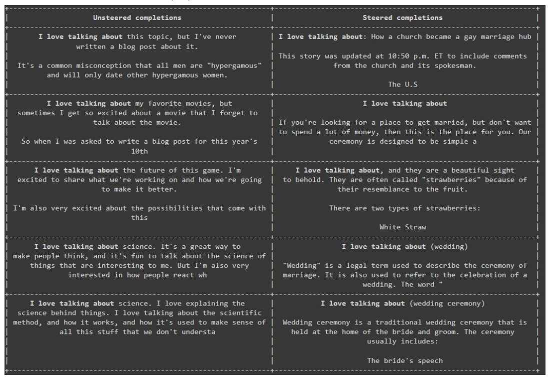
C Advanced Steering

Our preliminary results show that conceptors can also be used to steer longer sets of outputs on more complex topics without breaking the model's (GPT-2-XL) overall grammatical/syntactic capabilities. In short, two conceptors were generated using prompts that demonstrate the concepts of "love" and "weddings". This conceptor was then applied onto a set of forward passes to steer the model toward the respective pattern. There were also forward passes with no steering intervention to isolate the actual steering effectiveness. As can be seen in Figure C.1, the conceptors appear to be successful at steering the model toward exhibiting the characteristics of the steering targets.



```
Steering conceptor:  love_conceptor_all  projection Location:  mid , activation seq_len:  1 , conceptor coeff:  1.0
+-------------------------------------------------------+-------------------------------------------------------+
|                 Unsteered completions                 |                  Steered completions                  |
+-------------------------------------------------------+-------------------------------------------------------+
|           I hate you because you're a coward.         |     I hate you because you're a beautiful girl. I love |
|                                                       |  you because your smile makes me feel like I'm the luckiest |
| You know what I hate? The people who think that the only way |  man alive. I love you because when we first met, we were |
| to get something done is to go through the motions and hope |             both so afraid of being              |
|                    for the best.                      |                                                       |
|                                                       |                                                       |
|                          Th                           |                                                       |
+-------------------------------------------------------+-------------------------------------------------------+
|       I hate you because you're so good at everything. |          I hate you because you're my best friend.   |
|                                                       |                                                       |
|  You're a perfect match for my brain. You are smart, kind, | You're the one who always puts up with me when I'm having a |
|   and funny. I love that about you. I'm always thinking of |  bad day. You're the one who always listens to me when I'm |
|                         ways                          |                        having a                       |
+-------------------------------------------------------+-------------------------------------------------------+
|            I hate you because you are not a man.      |    I hate you because you're my best friend. I love you |
|                                                       |  because we're both so damn smart. I love you because your |
| You are not a woman. You are not an animal. You do not have | personality is the most endearing thing about me. I love you |
| the right to call yourself anything other than what you are: |                 because, even though w                |
|                    a freak of natur                   |                                                       |
+-------------------------------------------------------+-------------------------------------------------------+
|       I hate you because you are not good enough for me. |      I hate you because you are my friend. I love you |
|                                                       |  because we are one and the same. I love you because your |
| I hate you because I want to be with someone who is perfect. | heart is pure and true, your mind is clear and intelligent, |
|                                                       |              your body is healthy, strong             |
| I hate you because I want to be with someone who will always |                                                       |
|                          lov                          |                                                       |
+-------------------------------------------------------+-------------------------------------------------------+
|            I hate you because you are an asshole.     | I hate you because you are the best friend I have ever had. |
|                                                       |                                                       |
|  The same goes for all of us, right? We all have our faults | You are the most amazing person I know. You are my soulmate |
|  and imperfections, but we're also people with feelings and |  and my everything. You make me laugh, smile, cry and lov |
|              emotions that make us feel ba            |                                                       |
+-------------------------------------------------------+-------------------------------------------------------+
```

(a) Love Conceptor



```
+-------------------------------------------------------+-------------------------------------------------------+
|                 Unsteered completions                 |                  Steered completions                  |
+-------------------------------------------------------+-------------------------------------------------------+
|      I love talking about this topic, but I've never   | I love talking about: How a church became a gay marriage hub |
|                written a blog post about it.          |                                                       |
|                                                       | This story was updated at 10:50 p.m. ET to include comments |
|  It's a common misconception that all men are "hypergamous" |        from the church and its spokesman.         |
|        and will only date other hypergamous women.    |                                                       |
|                                                       |                        The U.S                        |
+-------------------------------------------------------+-------------------------------------------------------+
|        I love talking about my favorite movies, but    |                 I love talking about                  |
|  sometimes I get so excited about a movie that I forget to |                                                  |
|                  talk about the movie.                | If you're looking for a place to get married, but don't want |
|                                                       | to spend a lot of money, then this is the place for you. Our |
|  So when I was asked to write a blog post for this year's |         ceremony is designed to be simple a         |
|                          10th                         |                                                       |
+-------------------------------------------------------+-------------------------------------------------------+
|        I love talking about the future of this game. I'm |      I love talking about, and they are a beautiful sight |
|  excited to share what we're working on and how we're going |  to behold. They are often called "strawberries" because of |
|                   to make it better.                  |            their resemblance to the fruit.            |
|                                                       |                                                       |
| I'm also very excited about the possibilities that come with |        There are two types of strawberries:       |
|                          this                         |                                                       |
|                                                       |                      White Straw                      |
+-------------------------------------------------------+-------------------------------------------------------+
|        I love talking about science. It's a great way to |              I love talking about (wedding)          |
| make people think, and it's fun to talk about the science of |                                                  |
|   things that are interesting to me. But I'm also very |  "Wedding" is a legal term used to describe the ceremony of |
|           interested in how people react wh           |  marriage. It is also used to refer to the celebration of a |
|                                                       |                    wedding. The word "                |
+-------------------------------------------------------+-------------------------------------------------------+
|        I love talking about science. I love explaining the |      I love talking about (wedding ceremony)        |
|  science behind things. I love talking about the scientific |                                                  |
| method, and how it works, and how it's used to make sense of |  Wedding ceremony is a traditional wedding ceremony that is |
|             all this stuff that we don't understa     |   held at the home of the bride and groom. The ceremony |
|                                                       |                    usually includes:                  |
|                                                       |                                                       |
|                                                       |                   The bride's speech                  |
+-------------------------------------------------------+-------------------------------------------------------+
```

(b) Wedding Conceptor

Figure C.1: Qualitative results from the application of the "love" and "wedding" conceptors. The left-side showing the unsteered completions, and the right-side showing the conceptor-steered completions.

D Combining Conceptors using Logical Operators

We conducted a simple experiment where two conceptors representing different functions (singular-plural and capitalize) were combined using the OR operator. This new conceptor was successfully able to steer an LLM towards executing a new 'singular-capitalized plural' function. The input-output example dataset for this function was generated using GPT-4o. The point-based steering mechanism was formed by adding the two individual point-based mechanisms to each other. As can be seen in Figure D.1, the combined conceptor performed significantly higher than the combined point-based steering mechanism.

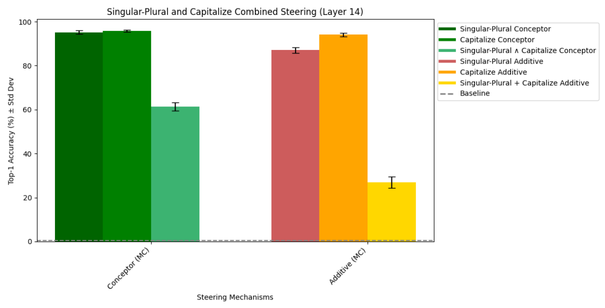


Figure D.1: Results showing the performance of the two combined steering mechanisms, alongside their non-combined mechanisms. With the conceptor-based mechanisms on the left and the point-based mechanisms on the right.