A STEERING METHOD AND EMBEDDING METHOD IMPLEMENTATIONS

A.1 DETAILED IMPLEMENTATION OF STEERING METHODS

Let f be a decoder-only language model with L layers and hidden size d. Each triplet comparison is denoted as $t_i = (x_{\text{ref},i}, x_{1,i}, x_{2,i})$ for $i = 1, \ldots, n$. Each prompt consists of a sequence of n triplets $[t_1, \ldots, t_n]$, serialized into a token sequence $x = [x_1, \ldots, x_T]$. The model produces hidden states $h_j^l \in \mathbb{R}^d$ at each token position x_j and layer l.

For all prompting, prompts are formatted as natural language strings of the form:

```
Choose the item that is most similar to the first item
in terms of <d>.  Respond only with the name of the
item exactly as written.
<x_ref> + <x_1> OR <x_ref> + <x_2>?
answer:  <x_ref> +
```

For steering methods, training examples consist of triplets without a natural language instruction like so:

```
<x_ref> + <x_1> OR <x_ref> + <x_2>?
<x_ref> + <x_answer>
```

Where examples are concatenated by commas. Each training example consisting of n triplets has a final incomplete training instance like so:

```
<x_ref> + <x_1> OR <x_ref> + <x_2>?
<x_ref> +
```

This "+" token is used to extract and steer representations for a corresponding "+" token in the zero-shot test example, which is identical to the final training example:

```
<x_ref> + <x_1> OR <x_ref> + <x_2>?
<x_ref> +
```

Fields are interpolated for some $d \in \{\textit{size}, \textit{kind}, \textit{neutral}\}$, triplet $t_n = (x_{\text{ref}}, x_1, x_2)$, and answer t_{answer} given the dimension d. We extract activations, logits, and apply all steering methods at the final input token $x_T =$ + in the last triplet t_n, depending on each method.

ZERO-SHOT PROMPT

In the zero-shot condition, the model is given a single triplet $t_n = (x_{\text{ref}}, x_1, x_2)$ and is asked to make a discrimination along a semantic dimension $d \in \{\textit{size}, \textit{kind}, \textit{neutral}\}$.

PROMPT WITH IN-CONTEXT EXAMPLES

In the in-context condition, the model is given a sequence of $n = 15$ complete triplets $[t_1, \ldots, t_{15}]$ and is asked to make a discrimination for the final triplet $t_{15} = (x_{\text{ref}}, x_1, x_2)$ along semantic dimension $d \in \{\textit{size}, \textit{kind}\}$.

TASK VECTOR

Following Hendel et al. (2023), we extract *task vectors* for the KIND and SIZE conditions by first constructing two prompts organized along each condition: x_{train}, containing 14 complete triplet examples and one final incomplete example (with "+"), and x_{test}, containing a single zero-shot incomplete triplet.

For each layer $\ell \in \{0, \ldots, L\}$, we extract the hidden activation in the residual stream at the final token position of x_{train} (i.e., the "+") and patch it into the corresponding position in x_{test}. The language model f then autoregressively generates a sequence $x_0, \ldots, x_k$ until a complete output is produced.

We repeat this procedure over 200 randomly generated $(x_{\text{train}}, x_{\text{test}})$ pairs, selecting the layer ℓ_d^* that yields the highest accuracy. Finally, using this optimal layer ℓ_d^*, we repeat the procedure across 2400 additional prompt pairs to generate task vector embeddings for both the SIZE and KIND conditions.

DIFFMEAN

DIFFMEAN constructs a steering vector by computing the average difference between latent representations of *positive* and *negative* examples along a target output dimension. Specifically, the mean latent representation of the positive examples is subtracted from that of the negative examples, producing a steering vector. This vector is then *added* (as opposed to task vectors, where it is patched) to the latent representation of a held-out prompt x_{test} in order to steer the model's output generation.

For a target steering dimension $d \in \{\textit{size}, \textit{kind}\}$ and its contrast d', we generate 15 triplet examples organized along d, and 15 along d', where the final triplet in each set is incomplete and ends with the `"+"` token.

Then, for a given layer $\ell \in \{0, \ldots, L\}$, we extract the residual stream representation r_T^ℓ at the final token position T from both $x_{\text{train},d}$ and $x_{\text{train},d'}$. The DIFFMEAN steering vector is then computed as the difference:

$$v_{\text{diff}} = r_T^\ell(x_{\text{train},d}) - r_T^\ell(x_{\text{train},d'})$$

This resulting vector v_{diff} defines a direction in the residual stream corresponding to the contrast between the dimensions $d \in \{\textit{size}, \textit{kind}\}$.

Similar to the task vector condition, We repeat this procedure over 200 randomly generated $(x_{\text{train}}, x_{\text{test}})$ pairs, selecting the layer ℓ_d^* that yields the highest accuracy. Finally, using this optimal layer ℓ_d^*, we repeat the procedure across 2400 additional prompt pairs to generate DiffMean embeddings for both the SIZE and KIND conditions.

SAEs

We use sparse autoencoders (SAEs) from GEMMASCOPE (Lieberum et al., 2024) to steer the model along interpretable directions in residual space. An SAE is a linear model that decomposes a residual stream vector $r \in \mathbb{R}^d$ as a sparse linear combination of features, where $W \in \mathbb{R}^{d \times k}$ is a learned feature dictionary and $z \in \mathbb{R}^k$ is a sparse activation vector. Each column of W defines a directional feature in residual space, and only a small number of features are active for any given input.

For each semantic dimension $d \in \{\textit{size}, \textit{kind}\}$, we construct 20 prompts and identify the feature $f \in \mathbb{R}^d$ with the highest average activation at layer $\ell = 20$. We steer the model by injecting $c \cdot f$ (with $c = 50$) into the residual stream at layer 20 (the only available layer for `gemma-2-9b-it` on GEMMASCOPE), and generate 2400 zero-shot completions from held-out prompts x_{test}. These completions are used to construct semantic embeddings for each SAE-steered dimension.

A.2 PROCRUSTES CORRELATIONS FOR ALL PROMPT AND STEERING METHODS

The comprehensive set of procrustes correlations for all steering methods, for `gemma-2-9b-it` and `gemma-2-27b-it`.

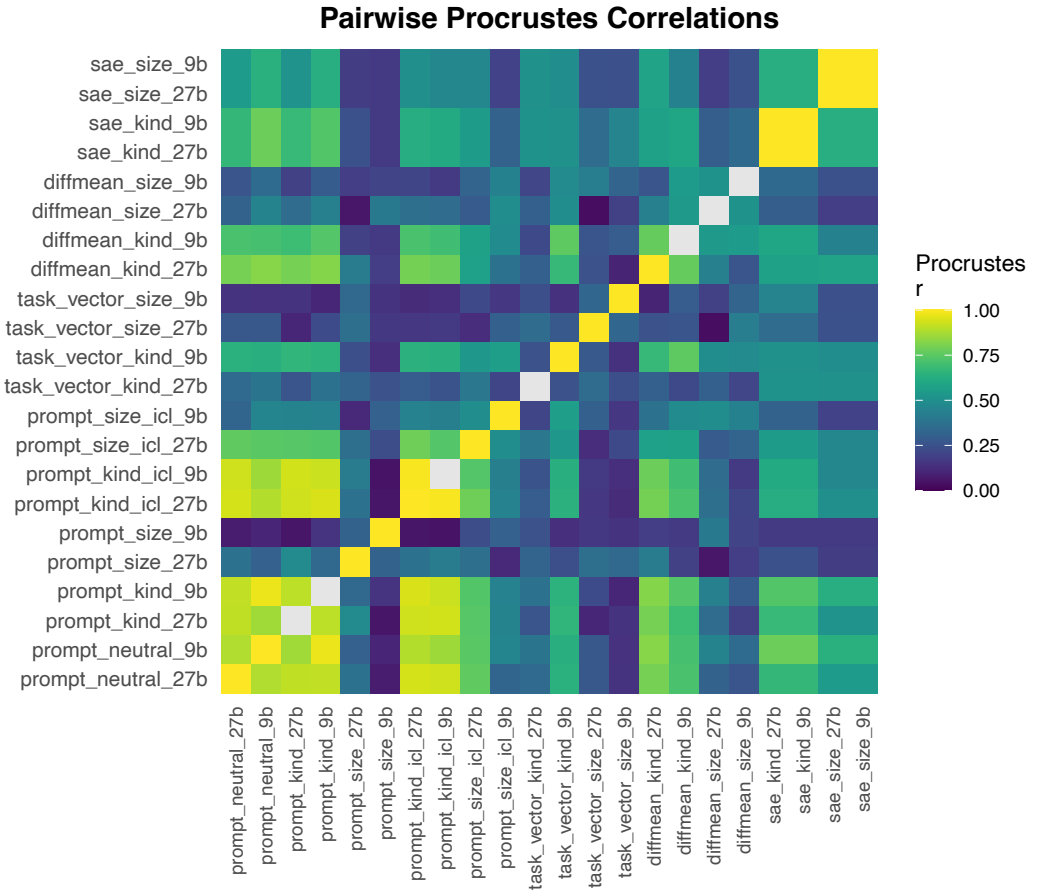
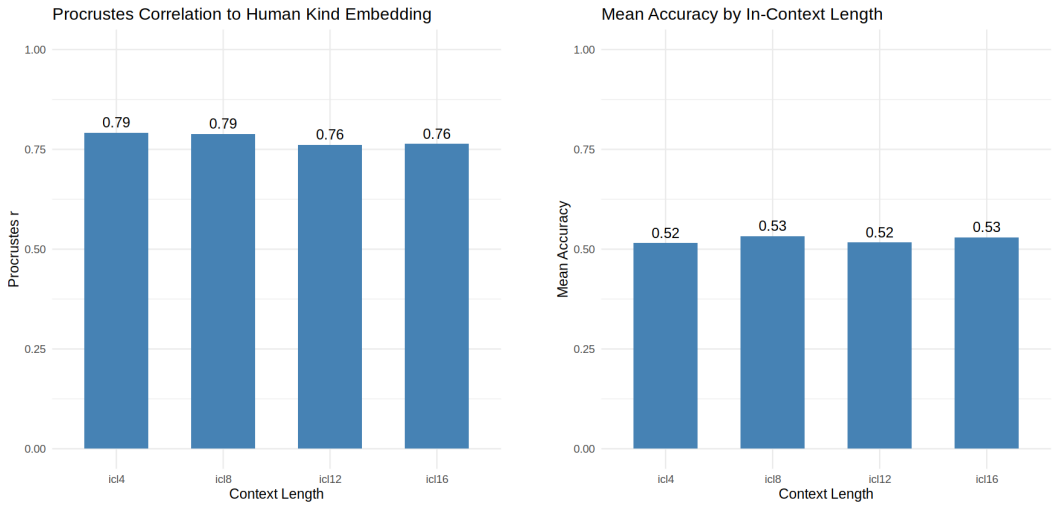


Figure 4: Full procrustes correlations for all methods

A.3 ICL Prompt Analysis

We systematically vary the number of example triplet pairs included in the KIND condition for `gemma-2-9b-it` to examine how changes to the input prompt affect model accuracy and human alignment. We find that there is no significant impact of the number of ICL examples on accuracy or alignment.



(a) Procrustes correlations with varied ICL examples

(b) Mean accuracy with varied ICL examples

Figure 5: Impact of varying the number of ICL examples on model performance

A.4 Embeddings Dimensions Analysis

Cumulative variance explained by the first k dimensions of the embeddings for `gemma-2-27b-it`, size condition. The first two dimensions of all embeddings were used for comparisons of representations.

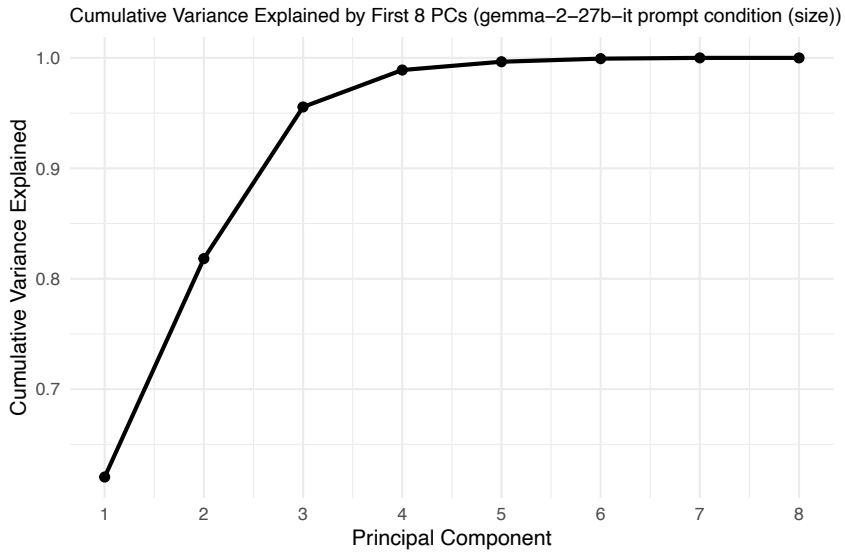
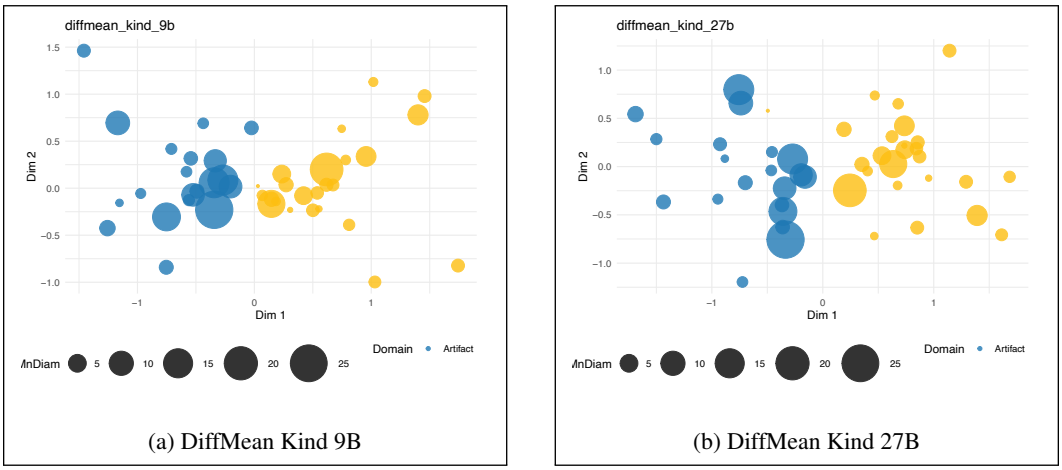


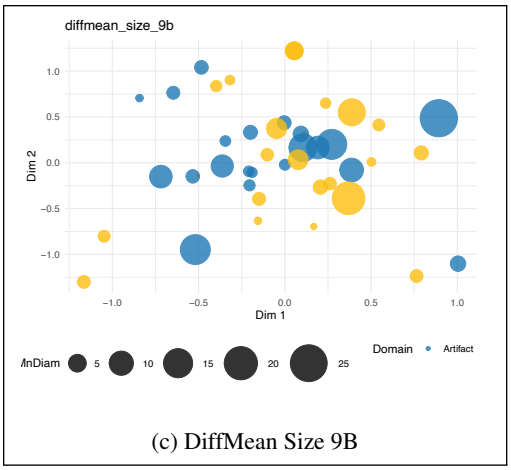
Figure 6: Cumulative variance explained by embedding dimensions

A.5 Full Embedding Plots

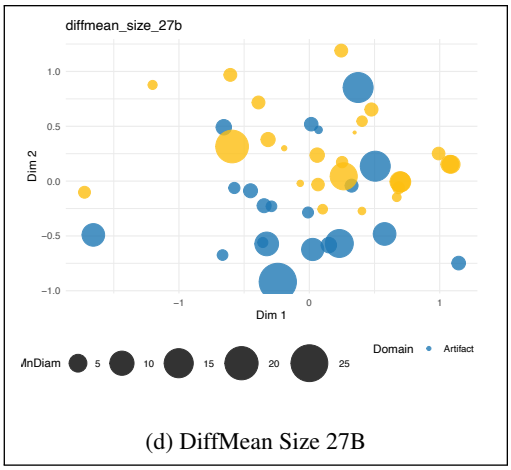


(a) DiffMean Kind 9B

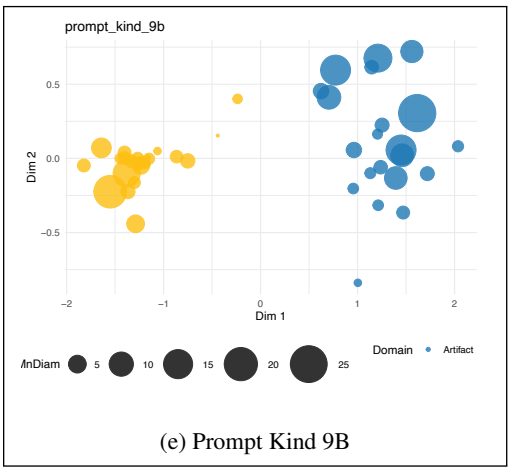
(b) DiffMean Kind 27B



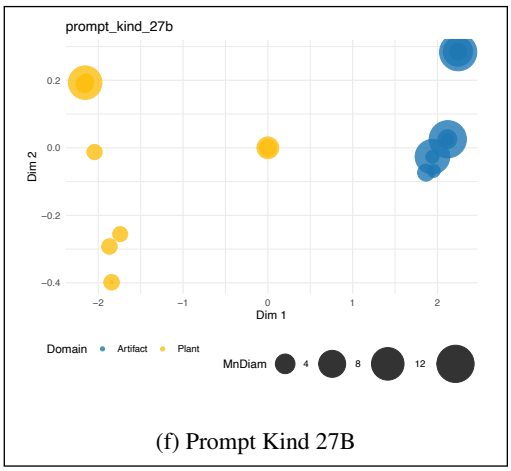
(c) DiffMean Size 9B



(d) DiffMean Size 27B

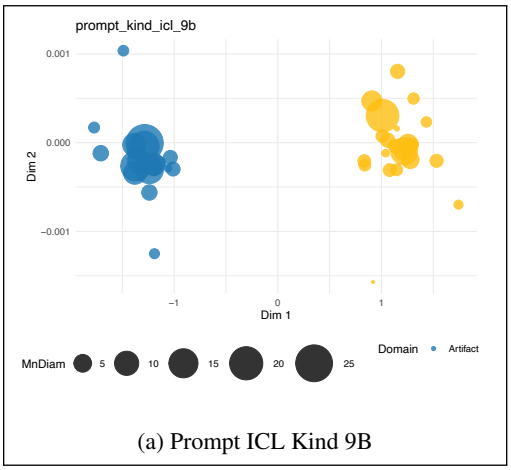


(e) Prompt Kind 9B

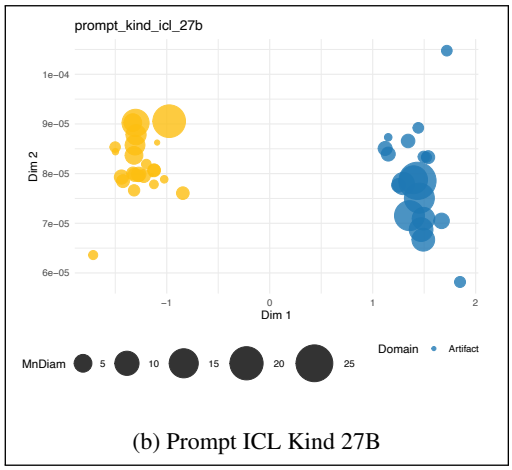


(f) Prompt Kind 27B

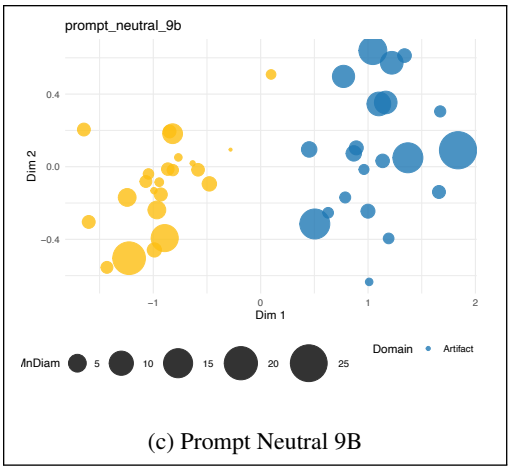
Figure 7: Embedding Plots: DiffMean and Prompt Methods



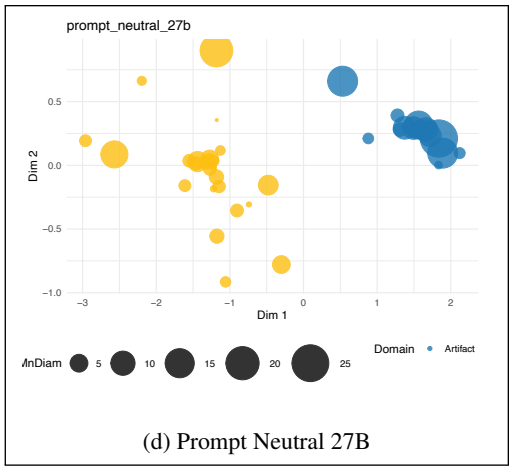
(a) Prompt ICL Kind 9B



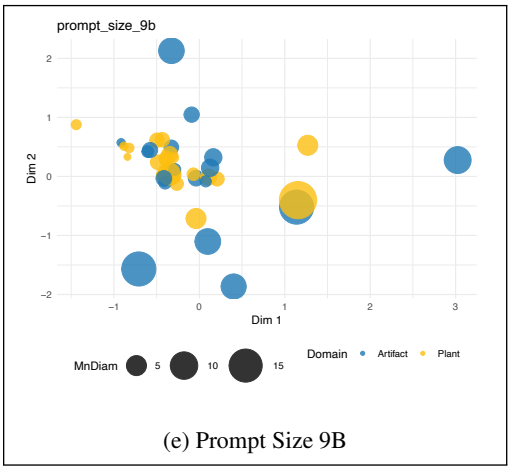
(b) Prompt ICL Kind 27B



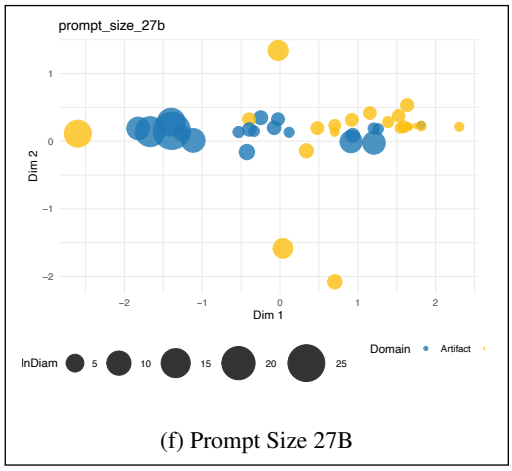
(c) Prompt Neutral 9B



(d) Prompt Neutral 27B

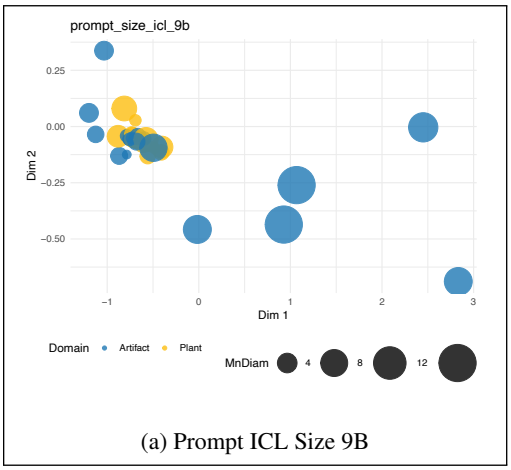


(e) Prompt Size 9B

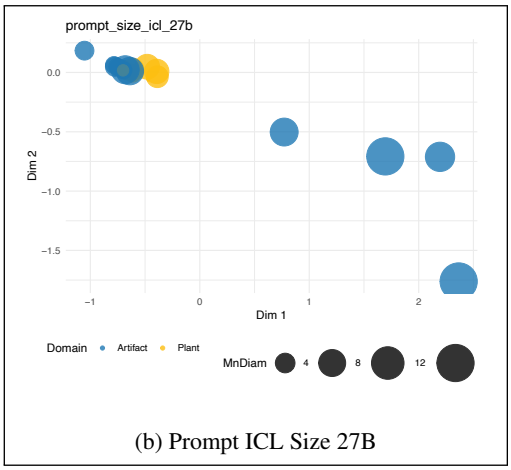


(f) Prompt Size 27B

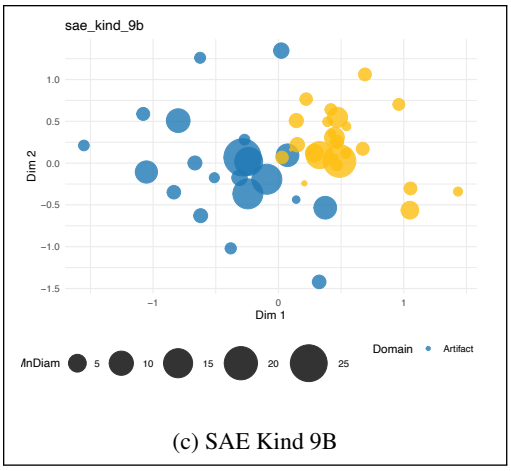
Figure 8: Embedding Plots: Prompt Method Variations



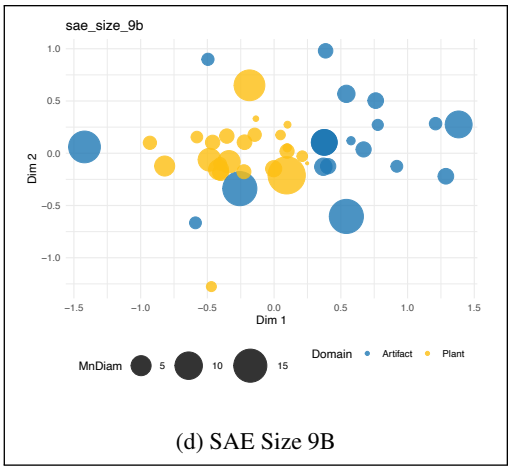
(a) Prompt ICL Size 9B



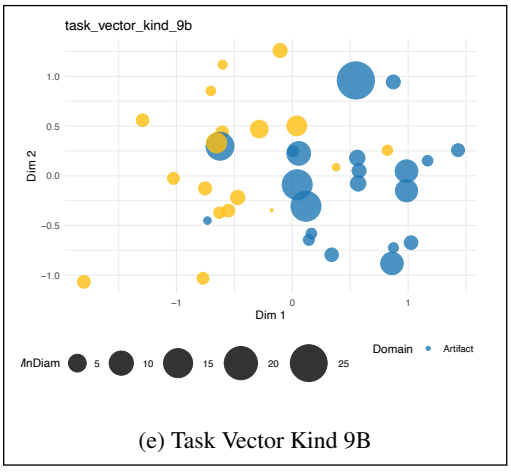
(b) Prompt ICL Size 27B



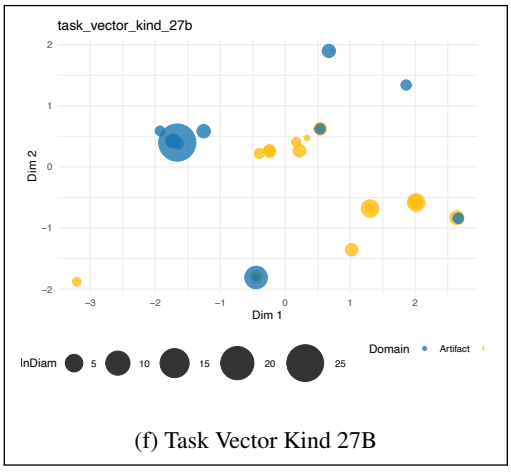
(c) SAE Kind 9B



(d) SAE Size 9B

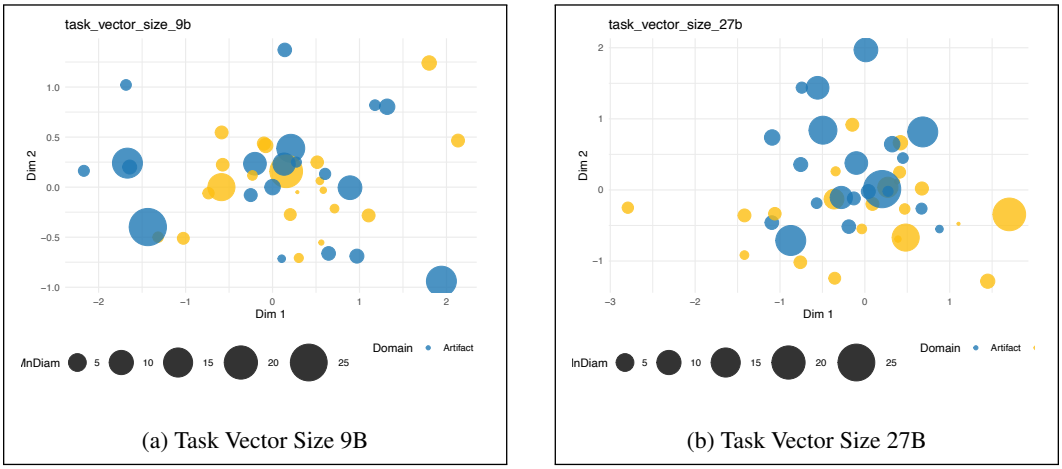


(e) Task Vector Kind 9B



(f) Task Vector Kind 27B

Figure 9: Embedding Plots: Prompt ICL, SAE, and Task Vector Methods



(a) Task Vector Size 9B

(b) Task Vector Size 27B

Figure 10: Embedding Plots: Task Vector Size Condition