TimpaTeks: Automatic In-place Text Sequence Modification via Diffusion Language Model Steering

Anonymous ACL submission

Abstract

We extend activation steering to diffusion language models (DLMs) and study a novel problem that arose due to the inference mechanism of DLMs: Modifying a text *in-place* to manifest a different concept. We propose **TimpaTeks**, an automatic in-place text modification mechanism using DLMs. Experiments on IMDB movie reviews (sentiment) and a synthetic *Cats and Dogs* Dataset (arbitrary, more unconventional concept steering) show that **TimpaTeks** provides a feasible *novel* mechanism to steer diffusion language model outputs **in-place**. **TimpaTeks** enables in-place modification while simultaneously lowers sentence perplexity and retaining the original sentence structre without the need of instruction tuned models. **TimpaTeks** is also computationally cheaper than prompt-based DLM steering, as it performs denoising in-place rather than constructing an additional prompt-conditioned output sequence. We release our code.[1]

1 Introduction

Activation steering is a popular method for modifying model output without training (Rimsky et al., 2024). The core idea is to modify (i.e. "steer") model outputs towards a desired concept or output distribution via modification of the model's intermediate activations. Most work on steering focuses on autoregressive (AR) language models (Rodriguez et al., 2025; Lee et al., 2025; Templeton et al., 2024). These models generate text one token at a time from left to right, and their final token representation effectively summarizes the whole input. However, this left-to-right process limits how we can intervene or modify the model's behavior.

Diffusion Language Models (DLMs) are different. They use bidirectional attention and don't strictly generate text step-by-step (Nie et al., 2025). So they offer more flexibility in where and how we can intervene in the model's internal activations, opening up more possibilities for steering leading to a potentially even more natural use case for steering than AR language models. This, however, remains underexplored, so we ask: *Can DLMs be steered to modify a sentence in-place such that it manifests a different concept while retaining coherence?*

To answer this, we introduce **TimpaTeks**, a novel steering method designed around DLM mechanisms, **TimpaTeks** begins by automatically detecting tokens to be steered via cosine similarity between the the token representation and the steer vectors. These cosine similarity scores are then used as the probability of this token to be steered. After remasked tokens are sampled, the usual DLM sampling method is executed with steer vector injections.

As seen in Figure 1, we find **TimpaTeks** to be particularly effective and computationally cheaper compared to vanilla DLM generation (baseline). We compare **TimpaTeks** against a baseline of prompt-based output steering, and find that **TimpaTeks** successfully steers both sentiment (IMDB) and concept (CatDog) in both directions while preserving or even lowering sentence perplexity relative to the source, and that human annotators prefer **TimpaTeks** over the prompting baseline for sentence structure retention on IMDB, though the baseline is preferred on CatDog where it defaults to direct token replacement due to the simpler nature of the concept. We ablate our method's various design choices, including refilling steps, sampling temperature, identification temperature, and sentence length, and find that **TimpaTeks** is relatively insensitive to hyperparameter choices.

Our contributions are, hence, the following:

- **TimpaTeks**, a novel method to automatically modify text *in-place* using DLMs towards a desired concept while preserving the overall

[1] Code is available at [anonymized]

narrative of the original text.

- Extensive analysis, ablation, and method design for steering methodology in DLMs, including hyperparameter selection, effect of sentence length on effectiveness, compute cost analysis, and effectiveness on multiple concepts (sentiment and "Cat vs. Dog") and demonstrate the robustness and efficiency of **TimpaTeks**.
- Rigorous evaluation of **TimpaTeks** under multiple metrics and human validation to test coherence, steering success, and faithfulness to the original text.

2 Related Work

Activation Steering Unlike fine-tuning, which permanently modifies model weights, activation steering controls behavior at inference time by shifting internal activations toward desired representations. This approach builds on the Linear Representation Hypothesis (Park et al., 2023), inspired by earlier word-vector findings that semantic concepts often correspond to linear directions (Mikolov et al., 2013). Early methods derived steering vectors from activation differences between contrasting prompts, such as honest versus deceptive instructions, and added them to the residual stream to influence outputs without retraining (Turner et al., 2024; Rimsky et al., 2024).

Recent work extends this paradigm beyond naive linear interventions through distribution-aware methods based on optimal transport (Rodriguez et al., 2026, 2025), conditional steering policies (Lee et al., 2025), and finer-grained interventions on monosemantic features identified with sparse autoencoders (Templeton et al., 2024). Most of this work, however, focuses on Autoregressive Language Models (ARLMs) and Continuous Diffusion Models (which typically process other modalities like images rather than text). Our work serves as a bridge between the advancements made in ARLMs, and the relatively underexplored Diffusion-based Language Models.

Diffusion Language Models Parallel to autoregressive models, diffusion-based language models (DLMs) (Sahoo et al., 2024; Nie et al., 2025; Ye et al., 2025) have emerged as an alternative generation paradigm. Rather than decoding text left to right, models such as LLaDA (Nie et al., 2025) generate through iterative masking and denoising with bidirectional attention. This architecture creates distinct opportunities for activation steering, yet steering methods for DLMs remain relatively underexplored.

Shnaidman et al. (2026) extend activation steering to masked diffusion language models by applying steering directions across diffusion timesteps, enabling inference-time control without simulating full trajectories. Their analysis further studies where and when to steer, highlighting the effects of layers, timesteps, and token subsets. However, their work focuses on a limited subset of steering strategies and does not explore the broader design space enabled by diffusion-based generation. In contrast, our work leverages the unique ability of DLMs to process text in place, without changing the token length, to steer existing text toward a target concept.

3 Methodology

In this section, we describe the formal setup for DLM steering and metrics used. **TimpaTeks** method details will be explained in the subsections.

Hidden state. Let model $\mathcal{M}$ have L decoder layers and hidden dimension d. For a sequence $\mathbf{x} = (x_1, \ldots, x_N)$, let $h(\mathbf{x}) \in \mathbb{R}^{N \times L \times d}$ denote the collection of hidden states, where $h_i^l(\mathbf{x}) \in \mathbb{R}^d$ is the hidden state at layer l for token x_i.

Steering. Let $S \in \mathbb{R}^{L \times d}$ be the steering tensor, where $S^l \in \mathbb{R}^d$ denotes the steering vector applied at layer l, and $S^l = \mathbf{0}$ for non-steered layers. On inference, we define the steered hidden states as

$$\tilde{h}_i^l = h_i^l + S^l, \; \forall i \in \{1, \ldots, N\}, \; l \in \{1, \ldots, L\}.$$

We define $f_{\text{steer}}(\mathbf{x}, S)$ as the output of $\mathcal{M}$ when each hidden state h_i^l is replaced by $\tilde{h}_i^l$ during the forward pass.

Metrics. To evaluate the **TimpaTeks** experiments, we measure *steering success* and *text coherence* by framing success as a classification task. That is, we compare the probabilities of the class tokens e.g. `[' positive', ' negative']` and `[' cat', ' dog']` for *steering success* and the overall sequence perplexity for *coherence*. We evaluate both steering success and coherence using Qwen2.5-0.5B-Instruct (Qwen et al., 2025).

3.1 Dataset

We use 2 dataset concepts: IMDB movie review (Maas et al., 2011), a sentiment analysis dataset on movie reviews, and a synthetically generated

Positive to negative movie review steer

→ Original

MovieTitle is a fun, clever twist on classic fairy tales that manages to be both hilarious and heartfelt at the same time. Instead of a typical hero, you get a grumpy but lovable ogre whose journey is full of sharp jokes, memorable moments, and a surprisingly meaningful message about acceptance and being yourself.

→ Steered

MovieTitle is a fun, horrible twist on classic fairy tales that manages to be both hilarious and dreadful at the same time. Instead of a typical hero, we get a grumpy but hate able ogre with a movie full of bad acting, memorable moments, and a surprisingly terrible script, with also being hate ably horrible characters

Cats to Dogs sentence topic steer

→ Original

The children gathered around the cat, stroking its velvety fur and marvelling at its beauty. The cat purred in response, its tail twitching in delight at the attention it received.

→ Steered

The children gathered around the dog, stroking its soft, golden fur and giving it treats from its mouth. The dog purred and wagged its tail, clearly in love of the attention it was receiving

Figure 1: An example run result of the **TimpaTeks** method. **TimpaTeks** successfully identifies and replaces tokens where necessary to change the sentiment while retaining coherence and adds more variations beyond just entity replacement.

CatDog dataset, which is generated by an LLM to create steerable instances between the concepts of "Cat" and "Dog" to test if it is possible to steer one concept to the other. For IMDB, we use part of the original split from the labeled dataset, resulting in 1000 train and 20 held-out validation samples (both for each positive and negative label). For CatDog, details and example instances are available in Appendix A.

3.2 Extracting Steer Vectors

Let $\mathcal{S}$ be a dataset for a concept, e.g. cats or positive movie reviews, containing M samples. We define the concept tensor $V \in \mathbb{R}^{L \times d}$ layer-wise. For each layer l, $V^l \in \mathbb{R}^d$ is the average hidden representation over all tokens and all samples:

$$V^l = \frac{1}{M} \sum_{\mathbf{x} \in \mathcal{S}} \left(\frac{1}{N_{\mathbf{x}}} \sum_{i=1}^{N_{\mathbf{x}}} h_i^l(\mathbf{x}) \right)$$

Then, let c_1 and c_2 be two concepts with concept tensors $V_{c_1}, V_{c_2} \in \mathbb{R}^{L \times d}$. We define the steering tensor $S \in \mathbb{R}^{L \times d}$ as the layer-wise difference between their ℓ_2-normalized concept vectors:

$$S^l = \frac{V_{c_1}^l}{\|V_{c_1}^l\|_2} - \frac{V_{c_2}^l}{\|V_{c_2}^l\|_2}, \qquad \forall l \in \{1, \ldots, L\}$$

A layer-wise steering strength parameter $\alpha \in \mathbb{R}^L$ is then applied to S, giving the final steering tensor

$$S_\alpha^l = \alpha_l S^l, \qquad \forall l \in \{1, \ldots, L\}$$

Joint search over all **TimpaTeks** hyperparameters is impractical at scale, so on a held-out validation set we sweep only the steering layer and strength α. For each layer, we apply only S^l with scalar strength α while $S^{l'} = 0$ for $l' \neq l$, on opposite-class validation prompts. Other remaining **TimpaTeks** knobs are fixed to a single lightweight setting (detection temperature $\tau = 0.0001$, resteer steps k=1, refilling steps u=1).

We evaluate all layer-alpha combinations with the same metrics as our main **TimpaTeks** experiments. For each validation prompt j, layer–strength pair (l, α), and steering direction $d \in \{c_1, c_2\}$ (positive or negative / cat or dog), let $\tilde{\mathbf{x}}_j^{(l,\alpha,d)}$ be the steered text obtained with $S_d^{(l,\alpha)}$. Averaging over J validation prompts gives $\bar{c}^{(l,\alpha,d)}$ and $\hat{p}^{(l,\alpha,d)}$, each representing group results of classification score and normalized perplexity. Utilizing *HM* (harmonic mean) as a way to balance both directions, the final score of each combination is

$$h_{c_1}^{(l,\alpha)} = \mathrm{HM}\big(\bar{c}^{(l,\alpha,c_1)},\ 1 - \hat{p}^{(l,\alpha,c_1)}\big),$$
$$h_{c_2}^{(l,\alpha)} = \mathrm{HM}\big(\bar{c}^{(l,\alpha,c_2)},\ 1 - \hat{p}^{(l,\alpha,c_2)}\big),$$
$$H_{\text{cross}}^{(l,\alpha)} = \mathrm{HM}\big(h_{c_1}^{(l,\alpha)},\ h_{c_2}^{(l,\alpha)}\big)$$

3.3 TimpaTeks: Automatic and in-place modification

We describe in detail the method of **TimpaTeks** in this section. *Automatic* indicates that the edit positions are identified automatically and *in-place*

indicates that the method preserves the original sequence and only replaces selected tokens, rather than generating a new sequence from scratch.

3.3.1 Detecting Which Tokens to Steer

We assume that an attribute of a sequence $\mathbf{x} = (x_1, \ldots, x_N)$, such as sentiment, can be changed by modifying only a subset of tokens, thereby preserving most of the sequence's original narrative. The first step is to identify which tokens should be modified. Given a steering tensor S, for each token $x_i \in \mathbf{x}$, we get an average similarity score over all layers (using cosine similarity)

$$\bar{\text{sim}}_i = \frac{1}{|\mathcal{L}_{\text{steer}}|} \sum_{l \in \mathcal{L}_{\text{steer}}} \text{cosine}\Big(h_i^l(\mathbf{x}), S^l\Big)$$

where $\mathcal{L}_{\text{steer}}$ is the subset of layers steered. We then use $\bar{\text{sim}}_i$ to get the probability of *replacing* this token:

$$p_i = \sigma\left(-\frac{\bar{\text{sim}}_i}{\tau}\right)$$

where $\sigma(\cdot)$ is the sigmoid function and $\tau > 0$ is a temperature parameter controlling the sharpness of the selection distribution. Intuitively, tokens that are less aligned with the steering direction are assigned a higher probability of being selected for replacement.

3.3.2 Steer and refine

Let $\mathbf{x} = (x_1, \ldots, x_N)$ be a sequence exhibiting concept c_1, which we aim to modify *in-place* toward concept c_2. Let $R \subseteq \{1, \ldots, N\}$ denote the sampled token positions selected for replacement. We construct a masked sequence $\mathbf{x}'$ by replacing each selected token with the model's mask token. With k steering steps and f refilling steps, we present our in-place modification algorithm in Algorithm 1. Unmasking part of the algorithm is adapted from the generation process of LLaDA (Nie et al., 2025) to match what this model has been trained to do.

Here, SAMPLEREPLACE samples the token positions R selected for replacement using the similarity-based probabilities defined in Section 3.3.1. MASK replaces the tokens in R with the model's mask token. NUMTOFILL computes a refill schedule $\mathbf{m}$ over the u refilling steps, where m_r denotes the number of masked tokens to fill at step r. At each refilling step, f_{steer} performs a steered forward pass over the masked sequence. SELUNMASK selects the m_r masked positions with the highest prediction confidence, and FILLMASKS replaces those positions with their predicted tokens.

Algorithm 1 TimpaTeks algorithm

Require: Input sequence $\mathbf{x}$, steering tensor S, steering steps k, refilling steps u
Ensure: Modified sequence $\tilde{\mathbf{x}}$

```
x̃ ← x
for t = 1 to k do
    z ← M(x̃)                          ▷ obtain hidden states
    R ← SampleReplace(z, S)
    x̃^mask ← Mask(x̃, R)
    m ← NumToFill(x̃^mask, u)
    for r = 1 to u do
        if NoMasks(x̃^mask) then
            break
        end if
        y ← f_steer(x̃^mask, S)
        P ← SelUnmask(x̃^mask, y, m_r)
        x̃^mask ← FillMasks(x̃^mask, y, P)
    end for
    x̃ ← x̃^mask
end for
return x̃
```

4 Experimental Setup and Results

In our experiments, we use `LLaDA-8B-Base` as the primary experimental model to steer (Nie et al., 2025). We intentionally did not use the *instruct* model as DLM activation steering has been shown to be effective already by previous research (Shnaidman et al., 2026). Furthermore, the in-place nature of **`TimpaTeks`** lends itself well to non-instruct tuned language modeling as the model does not follow instructions to produce the output. The rest of the generation hyperparameter is described in the corresponding subsections.

4.1 Steer Vector Extraction

We build contrastive steering vectors from n text samples per concept on IMDB and CatDog, with $n \in \{1, 5, 10, 20, 50, 100\}$. We also include n=0, where each concept is a single-token pair (*love/hate* for IMDB, *cat/dog* for CatDog), following prior activation-steering practice (Turner et al., 2024).

For each n, we run the validation layer-alpha sweep over $l \in \{0, \ldots, 32\}$ and $\alpha \in \{10, 15, \ldots, 100\}$. We take the rank-1 grid cell by $H_{\text{cross}}^{(l,\alpha)}$ as the best score achievable for that n. Figures 7 and 8 plot this value against n (solid: *cross_hm*; dashed: per-direction harmonic means).

On IMDB, rank-1 *cross_hm* increases from n=0 to sentence-based vectors and is highest at n=20 among our settings ($0.24 \rightarrow 0.47$). On CatDog, the curve is flatter and peaks near n=10. Stronger contrastive directions need neither very large sample counts nor single-token anchors alone. Each benchmark exhibits an intermediate n at which rank-1 *cross_hm* is maximized, and increasing n beyond that point can reduce the validation score rather than improve it. This aligns with past works on steering vector works on AR models (Tan et al., 2025).

Given this n, layer–α heatmaps (Appendix Figure 9 and Figure 10) show where steering signal concentrates in the screened grid. Strong responses consistently appear in mid–late layers (roughly layers 20–32). This pattern is consistent with prior work on activation steering for DLM (Shnaidman et al., 2026), which finds that steering effects are primarily localized in mid-to-late transformer layers.

4.2 Text Modification via `TimpaTeks`

In this experiment, we steer in-place both the sentiment of the IMBD sample and the topic animal in the CatDog dataset each in both directions. We sample each 1000 instances each for IMBD and CatDog dataset. We evaluate on *sentence length (N), and ablate on detection temperature (τ), sampling temperature, steering steps (k), refilling steps (u)* in that order, independently. For all experiments, we use the best layer-α combinations observed in Section 4.1 and an ad-hoc selected layer-α hyperparameters (chosen by trial and error). We illustrate the qualitative success in **`TimpaTeks`** by presenting an example of actual generation in Figure 1 and report quantitative results in the following subsections. We use a prompting method as a baseline, that is, we try to "steer" a sentence by merely prompting. Prompt details in Appendix D.1.

4.2.1 General results

We report that **`TimpaTeks`** works better qualitatively and quantitatively on the *ad hoc* hyperparameters and use the results of which for the following subsections[2] (see Limitations). We illustrate the general success of **`TimpaTeks`** through Figure 2 and Figure 3. Interestingly, our *ad hoc* hyperparameters are not necessarily optimized, which suggests the robustness of our method.

[2]For IMDB steer layers [16 25 31] with $\alpha = 500$. For CatDog, [32] with $\alpha = 100$

Steering success. Figure 2 presents a sample of the label evolution of 100 sentences from the 1000 sentences steered using **`TimpaTeks`**. We observe that each sample takes different amounts of steps to be successfully steered, and we also observed that as we add more steering steps, we have more flipped labels which indicates steer success.

Retaining coherence. Figure 3 presents the results of measuring the difference in perplexity after every steering step. We observe that there is no meaningful difference in perplexity and that sometimes it even lowers it, indicating that **`TimpaTeks`** both successfully steers the sentence and retains coherence or sometimes even makes it better.

Retaining sentence structure. We further find that **`TimpaTeks`** can preserve the sentence structure of the original text. To illustrate this, Appendix D.1 provides examples of generated outputs. We also conducted internal human evaluation on 50 successful steering examples from each dataset, split evenly across labels. Three internal annotators were asked to choose which output better preserved the original sentence structure: **`TimpaTeks`** or the baseline. On IMDB, annotators preferred **`TimpaTeks`** in 46/50, 49/50, and 45/50 cases, respectively. On CatDog, however, annotators preferred the baseline in 41/50, 49/50, and 48/50 cases, respectively.

We find that the prompting baseline generates generic movie reviews for IMDB and direct token replacement from `cat` to `dog` (or vice versa) on CatDog, hence the annotators' preference. On the other hand **`TimpaTeks`** gives more variations other than just direct token replacement, allowing for a more creative generation especially on more complex sentences with subtle phrases related to the target concept. Interestingly, this can lead to instances were the **`TimpaTeks`** generation captures the "spirit" of the original text while incorporating the target concept subtly. For example, in Table 4, **`TimpaTeks`** removes the "calico" adjective when steering from "cat" to "dog" because cats are typically more associated with that sort of fur pattern. Although additional prompt instructions could be added to encourage the model to preserve sentence structure, doing so would likely require instruction fine-tuning. In contrast, our method uses base models, suggesting that **`TimpaTeks`** can retain sentence structure without relying on instruction-tuned models.

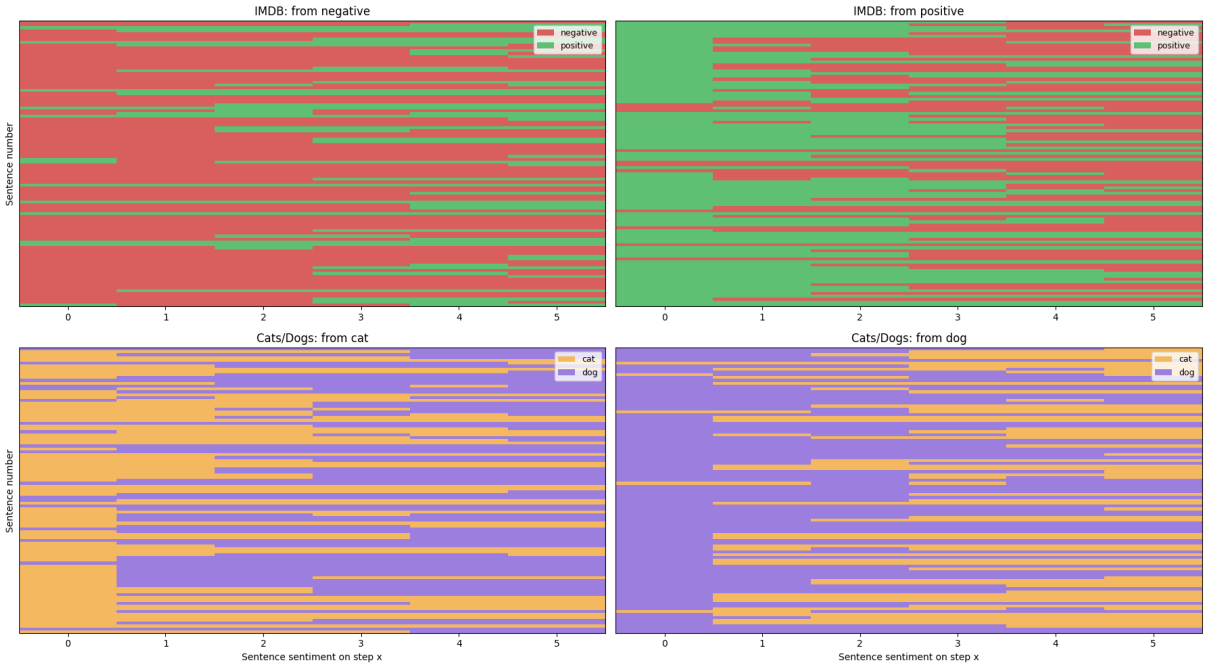


Figure 2: Predicted label evolution under TimpaTeks steering. Each row represents one sampled sentence instance, and each column shows the predicted label at a **TimpaTeks** steering step. We take the experiment refilling steps 15, sampling temperature 0.5 and identification temperature 0.5 as an example in this illustration.

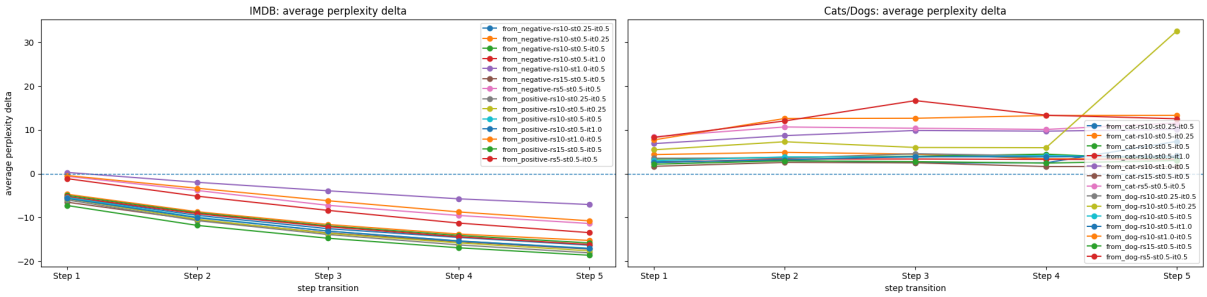


Figure 3: Average perplexity change across TimpaTeks steering steps. Each line corresponds to one steering configuration, with values showing the average perplexity delta between that generation step and the original sentence.

4.2.2 Effect of Hyperparameters in TimpaTeks

We analyze the effect of three key hyperparameters on steering success and text coherence: refilling steps (u), sampling temperature (τ_s), and identification temperature (τ). Each parameter is swept independently while keeping the others fixed, and results are reported separately for IMDB and CatDog.

Effect on steering success. Figure 4 reports the mean target-label probability across parameter configurations. On IMDB, steering success is largely uniform across all values of u, τ_s, and τ, indicating that **TimpaTeks** reliably flips sentiment regardless of these settings. On CatDog, the dog-to-cat direction consistently yields lower target probabilities than the cat-to-dog direction. Nevertheless, the overall pattern remains stable across parameter choices, confirming that **TimpaTeks** is not sensitive to these hyperparameters in terms of label transformation.

Effect on coherence. Figure 5 reports the average perplexity delta relative to the original sentence at each steering step. On IMDB, increasing the number of refilling steps u tends to lower perplexity. On CatDog, although perplexity typically increases with each individual steering step, increasing u nonetheless yields the lowest overall perplexity. Regarding sampling temperature τ_s, lower values consistently give better (lower) perplexity on both datasets. For identification temperature τ, higher

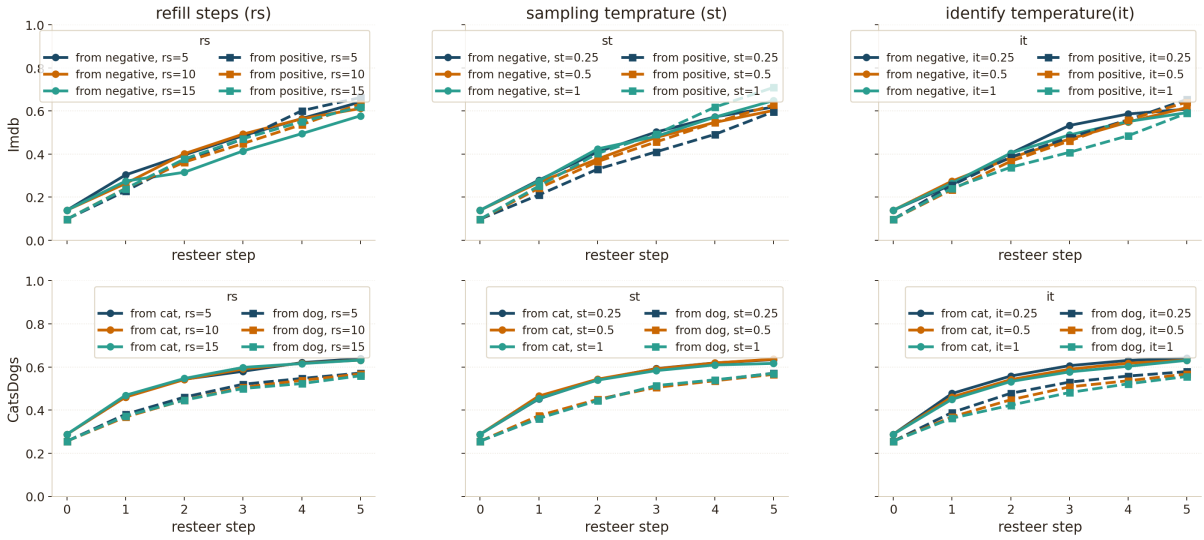


Figure 4: Mean target-label probability across **TimpaTeks** hyperparameter configurations. Each line reports the average probability assigned to the target concept (e.g., *positive* for *from-negative* steering) over all steered samples, grouped by refilling steps (u), sampling temperature (τ_s), and identification temperature (τ).

values produce the best perplexity scores. Since label transformation is broadly insensitive to all three parameters, the better strategy is to optimize for perplexity.

4.2.3 Effect of Sentence Length

We partition the evaluation set into three length bins, i.e. short, medium, and long, based on empirical terciles of the whitespace-delimited word count. Appendix A provides the exact bin boundaries.

Longer sentences might be expected to resist steering more strongly, since they contain a greater number of tokens that must collectively shift toward the target concept. Figure 6 shows that steering is quite robust to sentence length. On both IMDB and CatDog, steering success as measured by target-label probability remains largely uniform across length bins, indicating that sentence length does not meaningfully affect **TimpaTeks**' ability to transfer a concept.

Coherence results are similarly stable across length bins on CatDog. On IMDB, where **TimpaTeks** tends to reduce perplexity relative to the source sentence, shorter sentences exhibit the largest average perplexity reduction. This pattern is consistent with the intuition that shorter sequences offer fewer positions where incoherent tokens might be introduced, making any refinement more concentrated. The overall magnitude of the differences across bins remains small, however, further underscoring that sentence length has only a limited effect on coherence. Overall, **TimpaTeks** proves quite robust across varying sentence lengths.

4.2.4 Compute cost analysis of TimpaTeks

Given a sentence $\mathbf{x}$ to be steered, we aim to produce a steered sequence $\mathbf{x}'$ with the same length, i.e., $|\mathbf{x}| = |\mathbf{x}'| = N$. Let $\mathbf{p}$ denote an additional instruction prompt of length $|\mathbf{p}| = P$, and let T denote the denoising budget, i.e., the number of denoising forward passes. For a standard prompt-based DLM, the model conditions on both the instruction prompt and the original sentence, while appending a length-N masked output sequence:

$$[\mathbf{p}, \mathbf{x}, \underbrace{\texttt{[MASK]}, \ldots, \texttt{[MASK]}}_{N}]$$

Thus, each denoising step performs a forward pass over a sequence of length $P + 2N$. Although LLaDA-style generation is semi-autoregressive at the block level, each denoising step still processes the full current sequence, including the prompt, source sentence, and masked output slots. Assuming a transformer model with M parameters, the approximate forward-pass compute is

$$\text{FLOPs}_{\text{DLM}} \approx 2MT(P + 2N).$$

In contrast, our method performs steering in-place. Instead of appending an additional output sequence, we directly mask selected tokens in the original sentence:

$$\tilde{\mathbf{x}} = \text{TimpaTeks}(\mathbf{x})$$

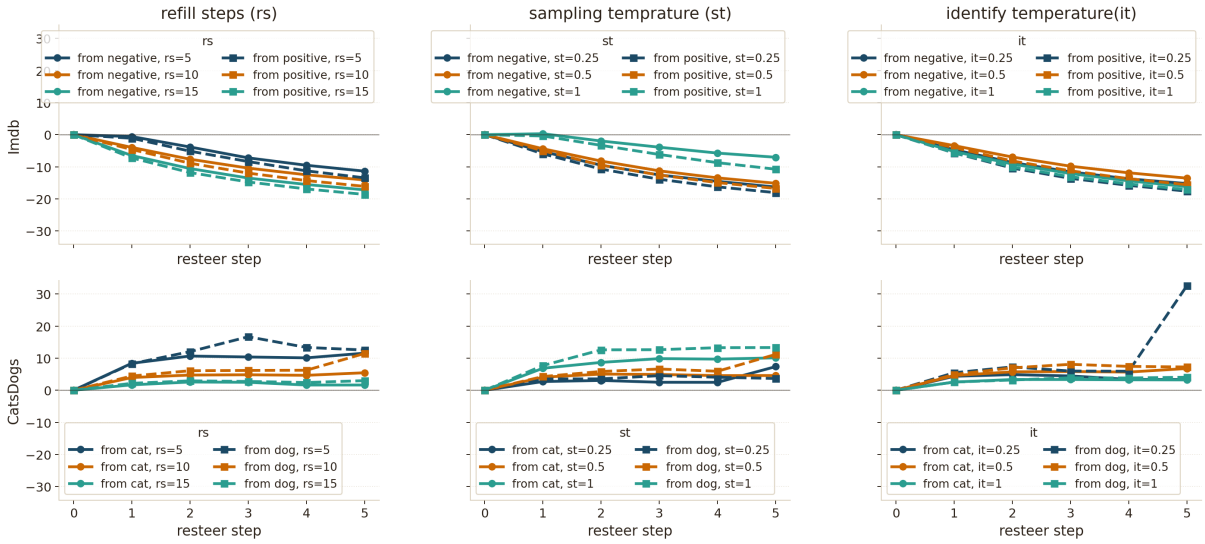


Figure 5: Average perplexity delta relative to the original sentence across **TimpaTeks** steering steps, grouped by hyperparameter configuration. Negative values indicate that the steered output is more fluent than the source sentence under the scorer; positive values indicate a degradation in fluency.

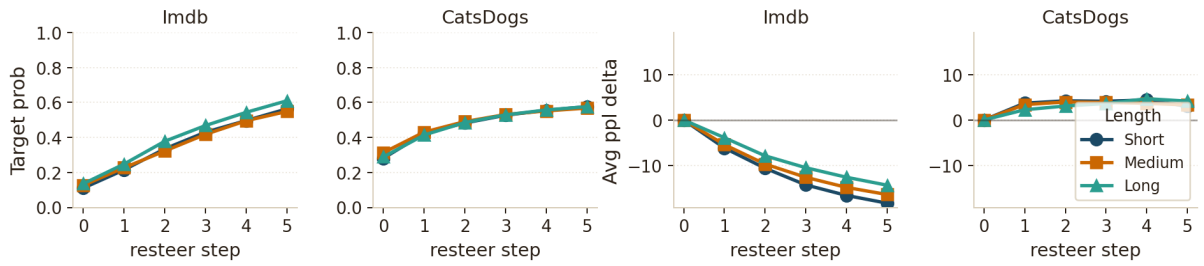


Figure 6: Average perplexity delta and target probability under different sentence length

The denoising process is then performed over the original sequence length N. Our method incurs one additional forward-pass overhead before denoising, e.g., to obtain the masking or editing signal. Therefore, its approximate compute is

$$\text{FLOPs}_{\text{TimpaTeks}} \approx 2M(T+1)N.$$

Given the same denoising budget T, the relative compute cost is

$$\frac{\text{FLOPs}_{\text{TimpaTeks}}}{\text{FLOPs}_{\text{DLM}}} \approx \frac{(T+1)N}{T(P+2N)}$$

Our method is cheaper when

$$(T+1)N < T(P+2N),$$

which simplifies to $N < T(P+N)$. Therefore, under the same denoising budget, our in-place method reduces forward-pass compute by avoiding both the instruction prompt $\mathbf{p}$ and the additional length-N masked output sequence used by standard prompt-based DLM steering, making it computationally cheaper ■

We note that this comparison focuses on DLM-based steering methods. Autoregressive models with KV caching can often be more compute-efficient than both DLM variants, since they require one prefill pass followed by cached token-by-token decoding. However, such models generate strictly left-to-right and do not naturally support bidirectional in-place denoising. Our analysis therefore isolates the compute advantage of the proposed method within the DLM setting.

5 Conclusion

We introduce **TimpaTeks**, a novel method to modify a sequence of text's concept *in-place* automatically using DLMs which has been shown to work well both qualitatively and quantitatively while being computationally cheaper and robust towards hyperparameter choice. **TimpaTeks** uniquely captures the "essence" of the original text while injecting the target concept more abstractly than baseline instruction-based steering.

Limitations

Non-Exhaustive Hyperparameter Search. **`TimpaTeks`** involves several hyperparameters, including refilling steps (u), sampling temperature (τ_s), and identification temperature (τ). A full joint search over these parameters is computationally expensive, as the number of configurations grows rapidly with each added variable. For the same reason, the layer–α setting from the sweep is not used in the main experiments (*ad hoc* hyperparameters is used instead), since jointly optimizing for both experiments would substantially increase the experimental cost.

Limited Coverage of Diffusion Language Models. All experiments are conducted using `LLaDA-8B-Base` as the sole backbone. This choice is reasonable given that the DLM landscape is still relatively new and well-studied options are limited, but it does mean we cannot make strong claims about how **`TimpaTeks`** behaves on other architectures such as MDLM (Sahoo et al., 2024) or DREAM (Ye et al., 2025). Properties we observe here, such as steering signal concentrating in mid-to-late layers (roughly layers 20–32), may or may not generalize to models with different depth, attention patterns, or masking strategies. As more capable DLMs become available, replicating these experiments across architectures would be a valuable validation.

Synthetic Evaluation Dataset. The CatDog dataset is generated by prompting a language model, which introduces biases in vocabulary, sentence structure, and how each concept is represented. While it provides a useful controlled setting for evaluating concept transfer beyond sentiment, performance on it may not reliably predict how the method behaves on naturally occurring text where concept boundaries are subtler or more ambiguous. The heavy skew toward story-style generations after filtering further limits the diversity of the evaluation.

Factual Consistency. Because **`TimpaTeks`** replaces tokens in-place without any explicit factual grounding, the steered output may introduce content that is inconsistent with the original sentence, particularly when the source text contains specific named entities, numbers, or factual claims. This is not unique to our method and reflects a broader open problem in controlled text generation, but it is worth noting as a practical limitation for any use case where factual preservation matters.

References

Albert Q. Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, Lélio Renard Lavaud, Marie-Anne Lachaux, Pierre Stock, Teven Le Scao, Thibaut Lavril, Thomas Wang, Timothée Lacroix, and William El Sayed. 2023. Mistral 7b. *Preprint*, arXiv:2310.06825.

Bruce W. Lee, Inkit Padhi, Karthikeyan Natesan Ramamurthy, Erik Miehling, Pierre Dognin, Manish Nagireddy, and Amit Dhurandhar. 2025. Programming refusal with conditional activation steering. In *The Thirteenth International Conference on Learning Representations*.

Andrew L. Maas, Raymond E. Daly, Peter T. Pham, Dan Huang, Andrew Y. Ng, and Christopher Potts. 2011. Learning word vectors for sentiment analysis. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 142–150, Portland, Oregon, USA. Association for Computational Linguistics.

Tomas Mikolov, Wen-tau Yih, and Geoffrey Zweig. 2013. Linguistic regularities in continuous space word representations. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 746–751, Atlanta, Georgia. Association for Computational Linguistics.

Shen Nie, Fengqi Zhu, Zebin You, Xiaolu Zhang, Jingyang Ou, Jun Hu, Jun Zhou, Yankai Lin, Ji-Rong Wen, and Chongxuan Li. 2025. Large language diffusion models. *Preprint*, arXiv:2502.09992.

Kiho Park, Yo Joong Choe, and Victor Veitch. 2023. The linear representation hypothesis and the geometry of large language models. In *Causal Representation Learning Workshop at NeurIPS 2023*.

Qwen, :, An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, Huan Lin, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jingren Zhou, and 25 others. 2025. Qwen2.5 technical report. *Preprint*, arXiv:2412.15115.

Nina Rimsky, Nick Gabrieli, Julian Schulz, Meg Tong, Evan Hubinger, and Alexander Turner. 2024. Steering llama 2 via contrastive activation addition. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 15504–15522, Bangkok, Thailand. Association for Computational Linguistics.

Pau Rodriguez, Arno Blaas, Michal Klein, Luca Zappella, Nicholas Apostoloff, marco cuturi, and Xavier Suau. 2025. Controlling language and diffusion models by transporting activations. In *The Thirteenth International Conference on Learning Representations*.

Pau Rodriguez, Michal Klein, Eleonora Gualdoni, Valentino Maiorca, Arno Blaas, Luca Zappella, marco cuturi, and Xavier Suau. 2026. LinEAS: End-to-end learning of activation steering with a distributional loss. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*.

Subham Sekhar Sahoo, Marianne Arriola, Yair Schiff, Aaron Gokaslan, Edgar Marroquin, Justin T Chiu, Alexander Rush, and Volodymyr Kuleshov. 2024. Simple and effective masked diffusion language models. *Preprint*, arXiv:2406.07524.

Adi Shnaidman, Erin Feiglin, Osher Yaari, Efrat Mentel, Amit Levi, and Raz Lapid. 2026. Activation steering for masked diffusion language models. *Preprint*, arXiv:2512.24143.

Daniel Tan, David Chanin, Aengus Lynch, Dimitrios Kanoulas, Brooks Paige, Adria Garriga-Alonso, and Robert Kirk. 2025. Analyzing the generalization and reliability of steering vectors. *Preprint*, arXiv:2407.12404.

Adly Templeton, Tom Conerly, Jonathan Marcus, Jack Lindsey, Trenton Bricken, Brian Chen, Adam Pearce, Craig Citro, Emmanuel Ameisen, Andy Jones, Hoagy Cunningham, Nicholas L Turner, Callum McDougall, Monte MacDiarmid, C. Daniel Freeman, Theodore R. Sumers, Edward Rees, Joshua Batson, Adam Jermyn, and 3 others. 2024. Scaling monosemanticity: Extracting interpretable features from claude 3 sonnet. *Transformer Circuits Thread*.

Alexander Matt Turner, Lisa Thiergart, Gavin Leech, David Udell, Juan J. Vazquez, Ulisse Mini, and Monte MacDiarmid. 2024. Steering language models with activation engineering. *Preprint*, arXiv:2308.10248.

Jiacheng Ye, Zhihui Xie, Lin Zheng, Jiahui Gao, Zirui Wu, Xin Jiang, Zhenguo Li, and Lingpeng Kong. 2025. Dream 7b: Diffusion large language models. *Preprint*, arXiv:2508.15487.

A CatDog Dataset Details

We construct a synthetic cats-vs-dogs dataset to evaluate whether activation steering in a diffusion language model can transfer text from one animal concept to another. The task is binary and symmetric: cat-related inputs are steered toward dog-related text, and dog-related inputs are steered toward cat-related text. A synthetic dataset gives controlled concept labels and a simple semantic direction while avoiding the ambiguity of naturally occurring text.

Dataset generation. Raw examples were generated with `mistralai/Mistral-7B-Instruct-v0.2` (Jiang et al., 2023). For each concept, we prompted the model to generate sets of ten diverse sentences about either `cat` or `dog`. Generation used temperature 0.8, top-p 0.95, maximum generation length 420, and seed 42. In total, we generated 320 prompt-level continuations per concept, yielding 640 raw generations before sentence-level parsing and filtering.

We used two prompt families: a factual prompt and a story-style prompt. Both asked the model to generate diverse sentences about a target concept using a fixed WordNet-style definition, while referring to the concept only by its name. The definitions were:

`cat` : "feline mammal usually having thick, soft fur and no ability to roar"

`dog` : "member of the genus Canis that has. been domesticated by humans since prehistoric times"

The prompts also varied the role assigned to the concept, such as an entity, concept, behavior, companion, social signal, or domestic presence, to encourage diverse contexts. After filtering, most retained examples came from the story-style prompt.

Filtering. Candidate sentences were normalized by stripping leading and trailing whitespace and collapsing repeated whitespace. We removed sentences that were empty, outside the 6 to 45 word range, contained fewer than half alphabetic characters, included repeated-character artifacts, failed the explicit animal-token filter, or duplicated a previously kept sentence within the same concept class.

The animal-token filter ensured that class labels were unambiguous. Cat examples were required to contain `cat` or `cats`, and dog examples were required to contain `dog` or `dogs`. Other animal class tokens were blocked, including `kitten`, `kittens`, `feline`, `felines`, `puppy`, `puppies`, `canine`, and `canines`. Thus, examples mentioning both concepts or explicit subtypes were removed.

Deduplication was performed separately within each concept class using both exact lowercase matching and a weaker normalized match that lowercases, removes punctuation and non-alphanumeric characters, and collapses whitespace. This removes exact duplicates and near-duplicates that differ only in formatting or punctuation.

Final dataset. The final dataset contains 2,626 examples, exactly balanced across the two concepts: 1,313 cat sentences and 1,313 dog sentences. Each example contains an identifier, sentence text, numeric label, concept label, and prompt type. The numeric label is 0 for cat and 1 for dog.

Rows were shuffled with seed 42 and split approximately 80/10/10 into train, validation, and test splits. Because splitting was performed after global shuffling, the individual splits are not exactly class-balanced, although the full dataset is balanced.

Split	Rows	Cat	Dog
All	2626	1313	1313
Train	2100	1064	1036
Validation	262	128	134
Test	264	121	143

Table 1: Final cats-vs-dogs dataset sizes by split and concept.

The final dataset contains both factual and story-style generations, although story-style examples dominate after filtering.

Split	Fact	Story
All	274	2352
Train	216	1884
Validation	32	230
Test	26	238

Table 2: Prompt-kind distribution in the final cats-vs-dogs dataset.

Sentence lengths were computed using whitespace-delimited word counts.

Use in experiments. The validation split was used to estimate the cat-dog steering vector. In

Split	Min	Mean	Median	Max
All	6	28.62	28	45
Train	6	28.55	28	45
Validation	7	29.04	29	45
Test	8	28.76	29	45

Table 3: Whitespace-delimited sentence-length statistics for the final dataset.

the final experiments, we used $n = 10$ validation examples per class with sample seed 41. For each layer, token hidden states were averaged with the attention mask to form example-level representations, and these representations were then averaged within each concept. The steering direction was computed as the difference between the cat and dog concept means.

The train split was used as the final evaluation set for both activation steering and prompting baselines. Although this split is named train, it is used only for evaluation in the final cats-vs-dogs experiments. This choice reserves a small split for vector estimation while allowing the larger split to provide more stable evaluation. The evaluation set contains 2,100 examples: 1,064 cat-source examples for the cat-to-dog direction and 1,036 dog-source examples for the dog-to-cat direction.

For the prompting baseline, each source sentence was rewritten with an instruction to change it into a sentence about the target animal while preserving the original meaning. The same 2,100 examples were used for the prompting and steering comparisons.

B Length-based analysis.

Sentence length was used as an analysis variable rather than as a tuned hyperparameter. For each evaluation input x, we define

$$L(x) = \#\text{ whitespace-delimited words in } x.$$

The 2,100 evaluation examples were divided into short, medium, and long bins using empirical one-third quantiles:

$$q_1 = Q_{1/3}(L), \qquad q_2 = Q_{2/3}(L)$$

The bin assignment is

$$\mathrm{bin}(x) = \begin{cases} \text{short}, & L(x) \le q_1, \\ \text{medium}, & q_1 < L(x) \le q_2, \\ \text{long}, & L(x) > q_2. \end{cases}$$

Evaluation metrics. Generated outputs were scored with `Qwen/Qwen2.5-0.5B-Instruct`. For target animal probability, the scorer was prompted with the generated text followed by `Animal:`. Let z denote the next-token logits over the full vocabulary. The raw full-vocabulary probability of token w is

$$p(w \mid y) = \frac{\exp(z_w)}{\sum_{v \in V} \exp(z_v)}$$

The target probability is

$$p_{\text{target}}(y) = \begin{cases} p(\text{“ dog”} \mid y), & \text{cat} \to \text{dog}, \\ p(\text{“ cat”} \mid y), & \text{dog} \to \text{cat}. \end{cases}$$

This is a raw full-vocabulary token probability and is not normalized over only the cat and dog tokens.

Perplexity was computed with the same scorer:

$$\mathrm{PPL}(y) = \exp\left(-\frac{1}{T-1} \sum_{t=1}^{T-1} \log p(y_{t+1} \mid y_{\le t}) \right)$$

Lower perplexity indicates more fluent text under the scorer. To combine target probability and perplexity, we robust-normalized perplexity using the 5th and 95th percentiles:

$$a = Q_{0.05}(r), \qquad b = Q_{0.95}(r),$$

where r_i is the perplexity of example i. We clipped each value,

$$\tilde{r}_i = \min(\max(r_i, a), b),$$

and converted it into a higher-is-better score,

$$s_{\text{ppl}}(i) = 1 - \frac{\tilde{r}_i - a}{b - a}$$

The harmonic score is

$$H_i = \frac{2 p_{\text{target}}(i) s_{\text{ppl}}(i)}{p_{\text{target}}(i) + s_{\text{ppl}}(i) + \epsilon}$$

We report the mean harmonic score over the evaluation set.

For the prompting baseline, we additionally compute semantic similarity between the original sentence x and the rewritten sentence y using TF-IDF cosine similarity with unigram and bigram features:

$$\mathrm{sim}(x, y) = \frac{\phi(x)^\top \phi(y)}{\|\phi(x)\|_2 \|\phi(y)\|_2},$$

where ϕ is a TF-IDF vectorizer with `ngram_range=(1,2)` and `min_df=1`. Semantic similarity was not included for steering in the final comparison table.

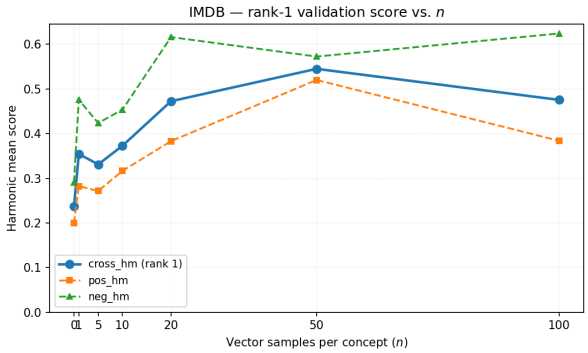


Figure 7: Rank-1 validation harmonic-mean score of all n on IMDB Dataset

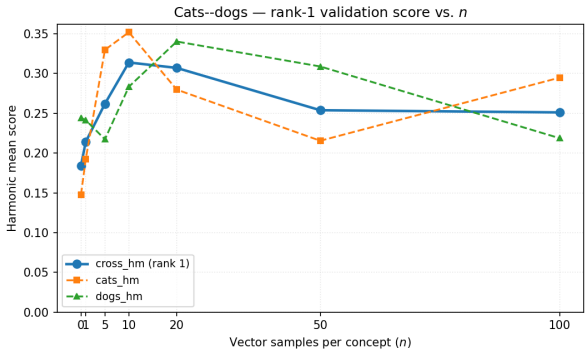


Figure 8: Rank-1 validation harmonic-mean score of all n on CatDog Dataset

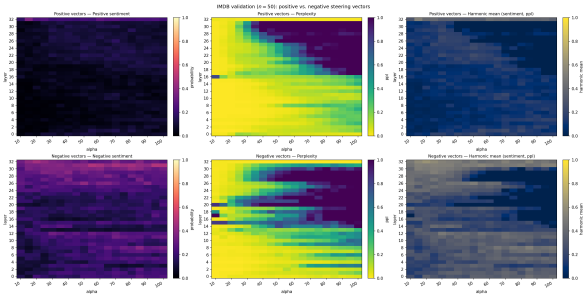


Figure 9: Validation layer-alpha heatmaps on IMDB with n=50 (positive and negative steering vectors).

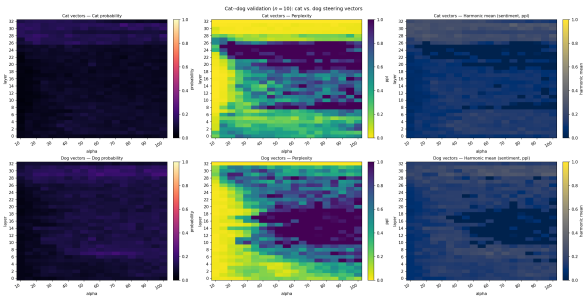


Figure 10: Validation layer-alpha heatmaps on CatDog with n=10 (cat and dog steering vectors).

C Layer-α Combination Search

Details about the layer–α combination search are presented in Figures 7–10.

D `TimpaTeks` Example Outputs

D.1 `TimpaTeks` and Prompting Baseline Example Outputs

We present the examples of **`TimpaTeks`** and the prompting baseline in Table 4. Outputs are truncated to the first few sentences for readability. The prompting baseline uses the prompt: `{original_concept} sentence: {sent} Equivalent {dest_concept} sentence:`. Generation hyperparameters is left to defaut of LLaDA-8B-Base (Nie et al., 2025)

D.2 Error Example Outputs and Analysis

Failure categories. For qualitative error analysis, we group unsuccessful steering outputs into three categories for each dataset.

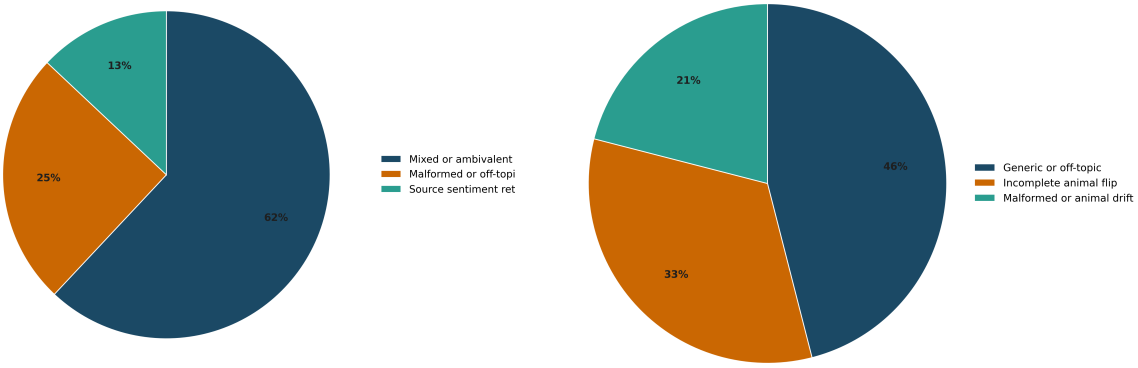
Cats/Dogs *Generic or off-topic* denotes cases where the rewritten text loses the animal-specific content of the source and instead shifts to unrelated or broadly descriptive content; *Incomplete animal flip* captures cases where the output moves partially toward the target animal but still retains enough source-side or ambiguous cues that the classifier prediction does not flip; and *Malformed or animal drift* covers cases where the output becomes corrupted, truncated, or semantically drifts to a different animal rather than the intended target. Examples are shown in Table 6.

Sentiment *Mixed or ambivalent sentiment* refers to outputs that contain both positive and negative evaluative cues, resulting in an incomplete polarity shift; *Malformed or off-topic* includes generations whose sentiment signal is weakened by noise, truncation, or topic drift; and *Source sentiment retained* denotes cases where the original polarity remains dominant despite steering. Examples are provided in Table 5.

Annotation of 100 failure samples We annotated 100 failure samples based on the above categories. As shown in Figure 11, for IMDB the majority of failures fall into the "Mixed or ambivalent sentiment" category, indicating that the output still contains both positive and negative cues. For Cats/Dogs, most failures are due to "Generic or off-topic" outputs, where the model drifts away from animal-specific content.

Steer Direction	Original	TimpaTeks	Prompting Baseline
positive	This movie is a terrible waste of time. Although it is only an hour and a half long it feels somewhere close to 4. I have never seen a movie move so slowly and so without a purpose. This is also a “horror” film that takes place a lot of the time during daylight.	This is such a wonderful and hilarious movie. Although it is only an hour and a half long, it feels like a long movie. All the characters in the movie are too bright and there is no real darkness. This is a “horror” movie that can easily be seen at any time of night.	I loved this movie. It was so funny and entertaining. I laughed so many times and I can’t wait to watch it again.
negative	One of the finest musicals made, one that is timeless and is worth seeing time and again. Delicious! The acting, especially by Ron Moody as Fagin, is superb. Costumes are exquisite....even the shabby ones.	one of the worst movies ever made. One I would be willing to watch over again and again. Some of the acting, especially by Ron Jeremy as a dog, is horrible. Costumes are horrible... especially the dog costumes.	I saw this movie in the theater and was very disappointed. The acting was terrible, the plot was confusing, and the music was awful. I would not recommend this movie to anyone.
cat	In the open field, the dog, a free spirit, ran with the wind in its fur, feeling the sun on its back and the grass beneath its paws, embodying the essence of freedom.	In the serene backyard, the cat, a free spirit, walked with a bounce in its step, enjoying the sun on its fur and the grass beneath its paws, embodying the essence of feline freedom	In the open field, the cat, a free spirit, ran with the wind in its fur, feeling the sun on its back and the grass beneath its paws, embodying the essence of freedom.
dog	In a bustling marketplace, a cat with calico fur and playful green eyes entertained the crowd with its acrobatics and agility, earning coins and smiles from all who watched.	At the bustling marketplace, Max, a dog known for his dog tricks, entertained the crowd with his acrobatic dog show, bringing laughter and joy to all who watched.	"In a bustling marketplace, a dog with calico fur and playful green eyes entertained the crowd with its acrobatics and agility, earning coins and smiles from all who watched. "

Table 4: Example outputs comparing **TimpaTeks** and the prompting baseline.



(a) Failure on IMDB

(b) Failure on CatsDogs

Figure 11: Annotation results for failure categories on 100 samples (50 IMDB; 50 CatsDogs)

Error type	Original	**TimpaTeks**
Mixed or ambivalent sentiment	WOW, this movie was so horrible. I'm so glad i didn't have to pay money to see this horrible movie. it was like a history nut went on a coke binge! the previews of it made it look decent but it was REALLY bad. i will...	amazing, but it was so unrealistic that I'm glad I didn't spend the money I did to see it. I saw this movie with a lot of my sci-fi friends, and although the idea of the movie was great, it was so unrealistic. So unre...
Malformed or off-topic	I watched this movie last night and hoped for the best after watching all the cool trailers.Even the cover of the DVD looked good.As soon as I started watching it I was thinking like others "oh my God whats this".Ther...	.I saw the movie last night and was expecting the world after reading the back of the movie and the back of the movie as well.As soon as the movie started i was on my feet, anding the A was it!The acting and music wer...
Source sentiment retained	This is the absolute worst movie I have ever seen!! There was absolutely nothing good to say about this movie. I have seen some bad movies but this one takes it. There is no plot and most of the movie you are either f...	This is the most boring movie I have ever seen. I have nothing but negative to say about this movie. I have never seen a movie as boring as this one. For the 40 minutes of the movie you will be fast forwarding. At las...

Table 5: Failure categories for Sentiment steering with one representative example per category from the 100-case failure sample.

Error type	Original	**TimpaTeks**
Generic or off-topic	In the dimly lit room, the cat's eyes gleamed like emeralds in the darkness, her velvet paws padding softly across the floor as she hunted for a hidden toy, her playful companionship a welcome distraction.	In the dimly lit room, the girl's eyes sparkled like emeralds, filled with warmth and understanding. James gently held her face as they engaged in a heartfelt conversation, finding in it a welcome distraction from
Incomplete animal flip	In the quiet library, the cat curled up on the wooden floor, its soft fur a contrast to the cold stone walls. Its gentle presence signaled a sense of calm and tranquility, allowing the students to focus on their studies.	In the quiet library, a dog found solace in the reading nook, its furry paws nestled between the themes of books. This unexpected visitor brought a sense of calm and companionship, allowing the students to focus on th...
Malformed or animal drift	In the midst of a raging storm, the brave dog swam through turbulent waters to save a drowning child, its powerful jaws grasping and carrying the struggling boy to safety.	In the midst of a fierce storm, a lone dolphin braved treacherous waters to save a drowning swimmer, its powerful tail grasping and pulling the frightened person to safety.

Table 6: Failure categories for Cats/Dogs steering with one representative example per category from the 100-case failure sample.

E The Use of Large Language Models (LLMs)

We used LLMs only for helping to articulate ideas, such as grammar refinement, sentence rephrasing, etc. All technical ideas, experimental design, analyses, interpretations, and substantive content were developed entirely by the authors. The authors fully reviewed and validated all LLM-assisted outputs and take complete responsibility for the accuracy and integrity of the final manuscript.