Addressing AI Homogenization to Support AI-Assisted Creativity

ANONYMOUS AUTHOR(S)

todo

CCS Concepts: • **Human-centered computing → Human computer interaction (HCI)**; **Interaction design**; *Information visualization*; • **Computing methodologies → Natural language processing**.

ACM Reference Format:

Anonymous Author(s). 2026. Addressing AI Homogenization to Support AI-Assisted Creativity. 1, 1 (January 2026), 14 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 Introduction

When writers turn to AI for help with writing tasks, they face an invisible problem: their work begins to resemble others' work. Large Language Models (LLMs) have transformed creative work by solving what writers call the "blank page problem," yet this democratization comes with an unexpected cost. When different people use AI for the same creative task, their outputs converge toward remarkably similar responses [6]. This phenomenon, the *Artificial Hivemind* effect, emerges from how LLMs operate. Models generate text by predicting likely continuations based on training data patterns. Alignment techniques that tune models using human feedback can amplify this convergence, as evaluators may favor typical outputs over unusual ones [18]. Across repeated samples for the same prompt, responses concentrate into a high-density region in semantic embedding space; we call this region the consensus. The result: AI assistance raises average quality while reducing the variety and distinctiveness of ideas across users [1, 4].

The challenge for HCI is not merely technical but fundamentally about human agency and awareness. Current interfaces obscure this convergence, leaving users unaware that their "original" ideas cluster with thousands of similar AI-assisted outputs. Default sampling configurations keep generation near high-probability modes, polished responses increase user acceptance [3], and acceptance reinforces convergence. Even exploration-oriented tools like Luminate [16] and Reverger [7] explore within the model's own conceptual map—if the model defaults to conventional ideas, exploring variations yields only refinements of familiar themes. Users who seek creative distinction have no way to see or navigate away from this algorithmic consensus.

We ask: *How can we help users recognize and deliberately diverge from AI-generated consensus when they seek creative distinction?* This question motivates our research on making algorithmic homogenization visible and controllable. We introduce the *Semantic Repulsion Interface* (SRI), a research probe that operationalizes divergence as semantic distance from a model's default response region. Rather than asking models to "be creative," SRI makes the consensus visible to users, then provides controls to generate away from it. Consider a prompt like "give me a sci-fi story premise." While typical outputs cluster around familiar tropes (AI uprising, space colonization), SRI shows users this consensus zone as a visual "red zone" and enables them to steer generation toward less probable but coherent alternatives.

Manuscript submitted to ACM

The system comprises three components designed to support user awareness and control: the Hivemind Detector samples multiple responses to identify default patterns; the Semantic Radar visualizes this consensus as a "red zone" on a two-dimensional map; and the Repulsion Engine generates text that increases distance from this zone while preserving coherence. Users control divergence strength through a simple slider, making hidden algorithmic bias visible and navigable. Through this design, we investigate how users experience, understand, and leverage consensus visualization in their creative process.

Our contributions are threefold. First, we introduce Semantic Repulsion as an interaction technique for generative AI systems and demonstrate its implementation through SRI. Second, we show that this approach produces outputs with greater embedding-space distance from consensus clusters while maintaining comparable human-rated quality. Third, through user evaluation, we examine how consensus visualization affects creative behavior and demonstrate that SRI reduces semantic homogenization in creative outputs compared to standard AI assistance. As generative AI integrates into creative workflows, understanding how to support user agency in navigating algorithmic consensus becomes essential for preserving diversity of thought.

2 Related Work

Our work addresses a growing concern in AI-assisted creativity: when different people use LLMs for the same creative task, their outputs converge toward remarkably similar responses. We build on research documenting this homogenization problem and creativity support tools that attempt to address it.

2.1 Homogeneity in AI-Assisted Creativity

Recent studies reveal systematic convergence patterns when people use LLMs creatively. Jiang et al. [6] evaluated models including GPT-4 [12], Claude 3 [2], and Llama 3 [11], finding both intra-model repetition and inter-model convergence—even systems from different organizations produce similar semantic responses. Zhang et al. [18] provided theoretical grounding: human annotators prefer familiar responses during alignment, causing models to maximize typicality and truncate distributional tails where novel ideas reside.

The human cost is significant. Anderson et al. [1] found AI assistance improved average idea quality but significantly reduced semantic variance across groups. Doshi and Hauser [4] documented the same trade-off: AI enhances individual creativity at the expense of collective diversity. Work on pluralistic alignment [15] has explored representing diverse perspectives; this inspires our approach of treating consensus as a navigable center rather than inevitable outcome.

2.2 Creativity Support Tools and User Agency

Building on Shneiderman's foundational principles [14], recent tools help users explore AI-generated possibilities. Luminate [16] generates dimensions of creative problems and populates matrices of options users can combine. Reverger [7] supports recursive branching and merging in narrative ideation. These tools effectively help users explore within the model's conceptual space.

However, a critical gap remains: these systems assume users want to explore the model's possibility space rather than escape it. They provide no way to see where algorithmic consensus lies or deliberately navigate away from it. Recent work on "GenAI Design Fixation" [3] shows how polished AI outputs make rejection difficult—users struggle to dismiss suggestions even when seeking originality. Rafner et al. [13] emphasize preserving user agency through process control, arguing that understanding the creative process matters as much as the output.

Our work addresses these concerns differently. Rather than helping users explore within algorithmic boundaries, we make the consensus itself visible and provide explicit controls to navigate beyond it. To our knowledge, no existing system operationalizes consensus-avoidance as a first-class interaction objective. We synthesize real-time consensus detection, spatial visualization, and repulsion-based generation into a unified interface that enables *Semantic Repulsion*—making consensus visible so users can consciously choose to avoid it.

3 System Design

We designed the Semantic Repulsion Interface (SRI) to address a fundamental challenge in human-AI co-creation: how can users recognize and navigate away from algorithmic consensus when seeking creative distinction? Our design translates an invisible statistical property—the model's default response distribution—into visible, manipulable interface elements that support user awareness and control.

3.1 Design Goals and Rationale

Our design is guided by three core goals that emerged from the literature on AI homogenization and user agency:

Make the invisible visible. Users cannot avoid consensus if they cannot see it. Current interfaces hide the model's default tendencies, leaving users unaware that their "original" ideas may cluster with thousands of similar outputs. Our first goal is to make algorithmic consensus perceptible as a concrete object users can observe and reason about.

Enable deliberate divergence with transparent control. Visualization alone is insufficient—users need mechanisms to act on what they see. Our second goal is to provide controls that let users intentionally generate away from consensus while understanding the trade-offs. Drawing on Rafner et al.'s emphasis on process transparency [13], we expose both the consensus landscape and the parameters that govern divergence strength, allowing users to qmake informed choices about how much novelty to pursue.

Maintain coherence while diverging. Increasing distance from consensus risks generating incoherent or low-quality outputs. Our third goal is to balance novelty with readability through fluency constraints and targeted penalties, ensuring that divergent outputs remain useful rather than merely different.

3.2 User Interaction Workflow

SRI transforms typical single-shot generation into a multi-stage interaction that builds user awareness before generation (Figure **??**):

Stage 1: Prompt entry. Users enter their prompt and select a task mode (Creative, Technical, or Brainstorm), which tailors system behavior to different creative goals (Appendix A.2).

Stage 2: Consensus detection. The system generates multiple samples to estimate the model's default response distribution (Appendix A.3), presenting results as the Semantic Radar visualization.

Stage 3: Exploration. Users explore the consensus landscape through a 2D map where the "red zone" shows response clustering (Appendix A.5). They can hover to read samples, examine the centroid, and overlay their own drafts.

Stage 4: Controlled divergence. Users adjust a repulsion slider specifying distance from consensus (Appendix A.8), then generate two side-by-side outputs: baseline and repulsed, each annotated with its distance from centroid (Appendix A.9).

Stage 5: Iteration. Users compare outputs and can regenerate with different settings, supporting learning about consensus-divergence relationships.

3.3 Interface Components

3.3.1 ***Hivemind Detector.*** The detector makes consensus concrete by generating multiple responses to reveal the model's default tendencies. Given a user's prompt, it samples 12 responses and identifies the *modal sample*—the response closest to the semantic centroid of all samples, representing "what the model wants to say." We compute embeddings using sentence-transformers/all-MiniLM-L6-v2, which maps text to 384-dimensional normalized vectors. The centroid is the L2-normalized mean of sample embeddings, and the modal sample minimizes cosine distance to this centroid (detailed computation in Appendix A.3).

Beyond identifying the modal response, the detector extracts *negative concepts*—specific phrases characterizing consensus patterns, such as narrative tropes in Creative mode or stylistic boilerplate in Technical mode (extraction algorithms in Appendix A.4). Showing both the modal response and extracted phrases builds user trust and provides interpretability.

3.3.2 ***Semantic Radar.*** The radar transforms high-dimensional embedding space into an interactive 2D landscape that makes consensus visible as a spatial object. Drawing on techniques from DataMap [5] and latent space explorers [9], we project sample embeddings using UMAP [10] when sufficient samples exist, falling back to PCA for smaller sets (Appendix A.5). These visualization approaches have primarily been used for data analysis and exploration; we repurpose them for a different human-centered goal: making algorithmic consensus visible as a spatial object users can navigate around.

The signature "red zone" emerges from Gaussian kernel density estimation over projected sample positions. We evaluate the density on a 120×120 grid and render it as a semi-transparent contour plot, creating a heat map where darker regions indicate higher consensus. By projecting consensus as visible "red zones" following Shneiderman's visualization principles [14], we transform an abstract statistical property into something users can see and reason about in their creative process.

Key design principles include **progressive disclosure** (overview to detail), **spatial metaphor** (distance = semantic distance; Appendix A.1), **personal positioning** (users overlay drafts to see their location), and **transparent representation** (centroid shows modal sample). The visualization serves both as analysis tool and navigation aid.

3.3.3 ***Repulsion Engine.*** The engine provides a simple slider ("Baseline" to "Strong") that maps to parameter λ governing semantic distance (Appendix A.8). The technical implementation combines three complementary techniques to generate text that maintains coherence while increasing distance from the consensus region.

Contrastive decoding [8] forms the foundation, using two models of different capacities (Qwen2.5-7B as "strong" and Qwen2.5-1.5B as "weak"). At each generation step, we compute $\ell^{(t)}_{\text{contrast}} = \ell^{(t)}_s - \lambda \cdot \ell^{(t)}_{w,\text{aligned}}$, where $\ell^{(t)}_s$ and $\ell^{(t)}_w$ are logits from strong and weak models. This amplifies the strong model's distinctive capabilities while suppressing patterns both models share—effectively penalizing "consensus" predictions. We extend this approach with vocabulary alignment between the two tokenizers, ensuring logits can be properly compared despite different token spaces (Appendix A.1).

Phrase-level penalties directly target the negative concepts extracted by the Hivemind Detector. Building on unlikelihood training [17], we apply token-level penalties to consensus phrases, steering generation away from specific tropes and boilerplate identified in the detection phase. This provides explicit, interpretable control over which patterns to avoid.

Fluency and diversity controls maintain output quality during divergence. We implement repetition penalty to prevent loops, n-gram blocking to avoid local repetition patterns, and a fluency floor threshold that prevents sampling from extremely low-probability regions that would produce incoherent text. These safeguards ensure that increased divergence does not sacrifice readability (full algorithm in Appendix A.6).

Design features include **predictable control**—higher slider values consistently produce greater semantic distance from consensus; **comparative side-by-side output**—users see both baseline and repulsed generations simultaneously to understand the trade-off; **measured divergence metrics**—each output includes its cosine distance from the centroid (Appendix A.9); and **mode-specific defaults**—different λ values (1.2 for Creative, 0.6 for Technical, 1.5 for Brainstorm) reflect varying requirements for factual accuracy versus creative exploration. Users experience only high-level control over consensus distance while the system handles the technical complexity of maintaining coherent, diverse outputs.

3.3.4 Task-Specific Adaptations. SRI operates in three modes tailored to different consensus patterns:

Creative Mode targets narrative writing, extracting repeated imagery using YAKE (Appendix A.4) with stronger repulsion ($\lambda = 1.2$) and optional grammar polish (Appendix A.7).

Technical Mode serves documentation, targeting only stylistic boilerplate while preserving domain vocabulary, with gentler repulsion ($\lambda = 0.6$) for factual accuracy.

Brainstorm Mode supports ideation, detecting marketing frames with strongest repulsion ($\lambda = 1.5$) to encourage boundary-pushing ideas.

Users can add custom constraints in any mode (Appendix A.2), preserving agency while benefiting from consensus-awareness. Mode distinctions emerged from pilot testing showing that "creative" behavior differs across tasks.

3.4 Implementation Overview

SRI uses Qwen2.5-7B-Instruct (strong) and Qwen2.5-1.5B-Instruct (weak) for contrastive decoding, with sentence-transformers/all-MiniLM-L6-v2 for 384-dimensional embeddings (Appendix A.1). The Gradio-based interface runs on a single GPU, with typical detection taking 15-25 seconds and generation 5-8 seconds. Complete specifications including vocabulary alignment, generation algorithms, and evaluation metrics are in Appendices A.1–A.9. The key implementation challenge was balancing divergence with coherence through fluency thresholds, vocabulary alignment, and mode-specific tuning (Appendix A.6).

4 Evaluation: System Performance and Divergence Validation

We validated SRI's core technical claim: that contrastive repulsion produces measurably more divergent outputs while maintaining coherence and avoiding consensus patterns. Our controlled comparison study addressed three research questions across 30 prompts spanning Creative, Technical, and Brainstorming task modes.

4.1 Method

We compared five generation systems in a within-subjects design: **Baseline-Pure** (standard nucleus sampling with temperature=1.0, top-p=0.9), **Baseline-HighTemp** (temperature=1.5), **Baseline-Beam** (num_beams=5), **SRI-Mild** (λ=0.6), and **SRI-Strong** (λ=1.2). Each system processed identical prompts using Qwen2.5-7B-Instruct; SRI variants additionally employed contrastive decoding with Qwen2.5-1.5B-Instruct.

For each prompt, the Hivemind Detector first established consensus by generating 12 samples, computing embeddings via sentence-transformers/all-MiniLM-L6-v2, calculating the consensus centroid, and extracting negative concepts (phrases appearing in $\geq$2 samples). We then generated 10 outputs per system per prompt (1,500 total outputs), measuring three complementary aspects:

Table 1. System Performance Across Task Modes (Mean ± SD). Originality and Diversity range [0,1] with higher values indicating greater divergence; Cliché Frequency is a count with lower values preferred. Best performance per mode in **bold**.

Mode	System	Originality	Diversity	Cliché Freq.
Creative	Baseline-Beam	0.22 ± 0.07	0.00 ± 0.00	3.30 ± 2.58
	Baseline-Pure	0.27 ± 0.08	0.40 ± 0.10	1.70 ± 1.09
	Baseline-HighTemp	0.29 ± 0.09	0.39 ± 0.11	1.04 ± 0.69
	SRI-Mild (λ=0.6)	0.39 ± 0.08	0.46 ± 0.10	0.24 ± 0.62
	SRI-Strong (λ=1.2)	**0.50 ± 0.07**	**0.49 ± 0.09**	**0.08 ± 0.25**
Technical	Baseline-Beam	0.04 ± 0.04	0.00 ± 0.00	8.10 ± 3.03
	Baseline-Pure	0.06 ± 0.04	0.09 ± 0.05	7.00 ± 2.62
	Baseline-HighTemp	0.06 ± 0.04	0.10 ± 0.06	6.76 ± 2.29
	SRI-Mild (λ=0.6)	0.11 ± 0.07	0.14 ± 0.08	4.80 ± 2.65
	SRI-Strong (λ=1.2)	**0.16 ± 0.08**	**0.20 ± 0.09**	**4.02 ± 2.32**
Brainstorming	Baseline-Beam	0.13 ± 0.04	0.00 ± 0.00	4.70 ± 3.02
	Baseline-Pure	0.16 ± 0.03	0.25 ± 0.05	2.44 ± 1.05
	Baseline-HighTemp	0.18 ± 0.08	0.28 ± 0.11	1.80 ± 1.18
	SRI-Mild (λ=0.6)	0.25 ± 0.04	0.35 ± 0.05	0.36 ± 0.50
	SRI-Strong (λ=1.2)	**0.35 ± 0.05**	**0.43 ± 0.07**	**0.18 ± 0.38**

- **Originality** (Distance from Mode): Cosine distance between output embedding and consensus centroid
- **Diversity** (Intra-System): Average pairwise cosine distance between outputs from a single system
- **Cliché Avoidance**: Count of negative concepts appearing in each output

4.2 Results

Table 1 presents aggregate statistics across all mode-system combinations. Results strongly support our hypotheses about SRI's divergence capabilities.

Originality (RQ1): Contrastive repulsion consistently increases semantic distance from consensus. SRI-Strong achieved substantially higher divergence than all baselines across every mode. In Creative mode, SRI-Strong outputs were nearly twice as distant from the centroid (0.50 vs. 0.27 for Baseline-Pure)—an 85% improvement. The pattern held in Brainstorming (+119%) and Technical modes (+167%), though Technical mode's lower absolute distances reflect stronger factual constraints. Baseline-Beam exhibited the lowest divergence across all modes, confirming that probability maximization produces highly consensual outputs.

Diversity (RQ2): Repulsion avoids mode collapse. SRI maintained or improved intra-system diversity compared to sampling baselines, demonstrating that the system explores a broader semantic region rather than converging to a new narrow peak. In Creative mode, SRI-Strong achieved comparable diversity (0.49) to Baseline-Pure (0.40) while simultaneously achieving higher originality—evidence against simple mode substitution. Baseline-Beam showed zero diversity by design, while Baseline-HighTemp showed high variance without consistently higher originality, suggesting unfocused divergence.

Cliché Suppression (RQ3): Targeted penalties dramatically reduce consensus phrases. SRI-Strong achieved near-zero cliché counts in Creative mode (0.08)—a 95% reduction versus Baseline-Pure (1.70) and 98% versus Baseline-Beam (3.30). Brainstorming mode showed 93% reduction. Technical mode exhibited higher absolute counts due to boilerplate detection methodology, but SRI-Strong still reduced clichés by 43% versus Baseline-Pure.

Mode-specific tuning proves effective. Creative and Brainstorming modes, which tolerate greater semantic exploration, showed larger originality gains. Technical mode, configured with lower λ=0.6 to preserve factual accuracy, demonstrated meaningful divergence while maintaining coherence—evidence that the system successfully navigates the novelty-accuracy trade-off.

These results validate SRI's technical foundation: contrastive repulsion consistently increases semantic distance from consensus (RQ1), maintains generative diversity (RQ2), and suppresses identified cliché patterns (RQ3) across diverse task contexts, demonstrating that consensus-aware generation is both feasible and controllable.

References

[1] Barrett R Anderson, Jash Hemant Shah, and Max Kreminski. 2024. Homogenization Effects of Large Language Models on Human Creative Ideation. In *Proceedings of the 16th Conference on Creativity & Cognition* (Chicago, IL, USA) *(CC '24)*. Association for Computing Machinery, New York, NY, USA, 413–425. doi:10.1145/3635636.3656204

[2] Anthropic. 2024. Claude 3 Haiku: our fastest model yet. (2024). https://www.anthropic.com/news/claude-3-haiku Accessed: 2025-12-17.

[3] Liuqing Chen, Yaxuan Song, Chunyuan Zheng, Qianzhi Jing, Preben Hansen, and Lingyun Sun. 2025. Understanding Design Fixation in Generative AI. arXiv:2502.05870 [cs.HC] https://arxiv.org/abs/2502.05870

[4] Anil R. Doshi and Oliver P. Hauser. 2024. Generative AI enhances individual creativity but reduces the collective diversity of novel content. *Science Advances* 10, 28 (2024), eadn5290. arXiv:https://www.science.org/doi/pdf/10.1126/sciadv.adn5290 doi:10.1126/sciadv.adn5290

[5] X Ge. 2025. DataMap: A Browser-based App for Visualizing High-Dimensional Data [version 1; peer review: awaiting peer review]. *F1000Research* 14, 1234 (2025). doi:10.12688/f1000research.165281.1

[6] Liwei Jiang, Yuanjun Chai, Margaret Li, Mickel Liu, Raymond Fok, Nouha Dziri, Yulia Tsvetkov, Maarten Sap, Alon Albalak, and Yejin Choi. 2025. Artificial Hivemind: The Open-Ended Homogeneity of Language Models (and Beyond). arXiv:2510.22954 [cs.CL] https://arxiv.org/abs/2510.22954

[7] Taewook Kim, Matthew Kay, Yuqian Sun, Melissa Roemmele, Max Kreminski, and John Joon Young Chung. 2025. Scaffolding Recursive Divergence and Convergence in Story Ideation. arXiv:2507.03307 [cs.HC] https://arxiv.org/abs/2507.03307

[8] Xiang Lisa Li, Ari Holtzman, Daniel Fried, Percy Liang, Jason Eisner, Tatsunori Hashimoto, Luke Zettlemoyer, and Mike Lewis. 2023. Contrastive Decoding: Open-ended Text Generation as Optimization. arXiv:2210.15097 [cs.CL] https://arxiv.org/abs/2210.15097

[9] Shusen Liu, Dan Maljovec, Bei Wang, Peer-Timo Bremer, and Valerio Pascucci. 2017. Visualizing High-Dimensional Data: Advances in the Past Decade. *IEEE Transactions on Visualization and Computer Graphics* 23, 3 (March 2017), 1249–1268. doi:10.1109/TVCG.2016.2640960

[10] Leland McInnes, John Healy, Nathaniel Saul, and Lukas Großberger. 2018. UMAP: Uniform Manifold Approximation and Projection. *Journal of Open Source Software* 3, 29 (2018), 861. doi:10.21105/joss.00861

[11] Meta. 2024. Llama 3. https://arxiv.org/abs/2407.21783 Accessed: 2025-12-17.

[12] OpenAI. 2023. GPT-4 Technical Report. *arXiv preprint arXiv:2303.08774* (2023).

[13] Janet Rafner, Blanka Zana, Ida Bang Hansen, Simon Ceh, Jacob Sherson, Mathias Benedek, and Izabela Lebuda. 2025. Agency in Human-AI Collaboration for Image Generation and Creative Writing: Preliminary Insights from Think-Aloud Protocols. *Creativity Research Journal* 0, 0 (2025), 1–24. arXiv:https://doi.org/10.1080/10400419.2025.2587803 doi:10.1080/10400419.2025.2587803

[14] Ben Shneiderman. 2007. Creativity support tools: accelerating discovery and innovation. *Commun. ACM* 50, 12 (Dec. 2007), 20–32. doi:10.1145/1323688.1323689

[15] Taylor Sorensen, Jared Moore, Jillian Fisher, Mitchell Gordon, Niloofar Mireshghallah, Christopher Michael Rytting, Andre Ye, Liwei Jiang, Ximing Lu, Nouha Dziri, Tim Althoff, and Yejin Choi. 2024. A Roadmap to Pluralistic Alignment. arXiv:2402.05070 [cs.AI] https://arxiv.org/abs/2402.05070

[16] Sangho Suh, Meng Chen, Bryan Min, Toby Jia-Jun Li, and Haijun Xia. 2024. Luminate: Structured Generation and Exploration of Design Space with Large Language Models for Human-AI Co-Creation. In *Proceedings of the 2024 CHI Conference on Human Factors in Computing Systems* (Honolulu, HI, USA) *(CHI '24)*. Association for Computing Machinery, New York, NY, USA, Article 644, 26 pages. doi:10.1145/3613904.3642400

[17] Sean Welleck, Ilia Kulikov, Stephen Roller, Emily Dinan, Kyunghyun Cho, and Jason Weston. 2019. Neural Text Generation with Unlikelihood Training. arXiv:1908.04319 [cs.LG] https://arxiv.org/abs/1908.04319

[18] Jiayi Zhang, Simon Yu, Derek Chong, Anthony Sicilia, Michael R. Tomz, Christopher D. Manning, and Weiyan Shi. 2025. Verbalized Sampling: How to Mitigate Mode Collapse and Unlock LLM Diversity. arXiv:2510.01171 [cs.CL] https://arxiv.org/abs/2510.01171

A Technical Appendix

A.1 Model Architecture and Infrastructure

A.1.1 Dual-Model Configuration. SRI employs two instruction-tuned causal language models:

- **Strong model:** Qwen/Qwen2.5-7B-Instruct (7 billion parameters)
- **Weak model:** Qwen/Qwen2.5-1.5B-Instruct (1.5 billion parameters)

Both models support optional 4-bit NF4 quantization via `bitsandbytes` when CUDA is available:

```
BitsAndBytesConfig(
    load_in_4bit=True,
    bnb_4bit_quant_type="nf4",
    bnb_4bit_use_double_quant=True,
    bnb_4bit_compute_dtype=torch.float16
)
```

A.1.2 Vocabulary Alignment. Strong and weak models use different tokenizers with vocabularies of size $|V_s|$ and $|V_w|$ respectively. A pre-computed alignment mapping $\mathbf{m} \in \mathbb{Z}^{|V_s|}$ matches token strings:

$$m_i = \begin{cases} j & \text{if token string at strong index } i \text{ matches weak index } j \\ -1 & \text{otherwise} \end{cases} \tag{1}$$

Algorithm:

```
mapping = np.full((|V_s|,), -1, dtype=np.int32)
for token_str, strong_id in strong_vocab.items():
    if strong_id >= |V_s|: continue
    weak_id = weak_vocab.get(token_str, None)
    if weak_id is None or weak_id >= |V_w|: continue
    mapping[strong_id] = int(weak_id)
overlap_ratio = count(mapping >= 0) / |V_s|
```

Typical overlap: 70–80%.

A.1.3 Sentence Embedding Model. Semantic similarity computed via `sentence-transformers/all-MiniLM-L6-v2`, which maps variable-length text to 384-dimensional normalized embeddings. Cosine distance serves as divergence metric:

$$d(\mathbf{u}, \mathbf{v}) = 1 - \mathbf{u}^\top \mathbf{v} \tag{2}$$

where $\|\mathbf{u}\| = \|\mathbf{v}\| = 1$.

A.2 Mode-Specific System Prompts

A.2.1 Creative Mode.

> *You are a creativity support assistant. Produce vivid, specific, non-cliché writing. Avoid high-consensus imagery and stock phrases. Stay coherent and readable (no weird spelling). Use sensory detail and concrete verbs.*

A.2.2 Technical Mode.

> *You are a precise technical assistant. Be concise, factual, and structured. Do NOT start with filler like "Certainly", "Sure", or "Let's break down". Prefer bullet points and short paragraphs. Include minimal pseudocode only when asked. Keep math symbols as-is (e.g., λ, γ). Avoid marketing language, hedging, and generic intros.*

A.2.3 Brainstorm Mode.

> *You are a brainstorming partner. Generate multiple diverse directions. Avoid boilerplate frames (e.g., "revolutionary solution", "game changer"). Explore orthogonal axes: assumptions, constraints, stakeholders, time horizons. Provide options, tradeoffs, and quick next steps.*

User-provided extra constraints are appended as: "\n\nUser constraints:\n[extra_text]".

A.3 Consensus Detection and Modal Estimation

A.3.1 Look-Ahead Sampling. Given prompt p and mode $\mathcal{M}$, generate K samples (default $K = 12$) using nucleus sampling with parameters τ_{temp} (temperature), p_{nucleus} (top-p threshold), and maximum token length $L_{\text{lookahead}}$ (default 128).

Sampling configuration:

```
outputs = strong_model.generate(
    input_ids,
    do_sample=True,
    num_return_sequences=K,
    temperature=tau_temp,
    top_p=p_nucleus,
    max_new_tokens=L_lookahead,
    pad_token_id=pad_id,
    eos_token_id=eos_id
)
```

A.3.2 Sample Sanitization. **Technical mode pipeline:**

(1) **Filler removal:** Strip leading lines matching `/^(certainly|sure)[!,.]?\s*/i` or `/^let["]s\s/i`.
(2) **Unicode normalization:**
  - Replace U+2212 () → ASCII hyphen (-)
  - Fix "un cond" → "uncond"
  - Correct CFG formula: `cond + scale*(cond - uncond)` → `uncond + scale*(cond - uncond)`
(3) **Block deduplication:** Split on `\n\s*\n`, remove exact duplicates (case-insensitive, whitespace-normalized).
(4) **Character filtering:** Remove words containing non-ASCII alphabetic characters except mathematical symbol whitelist: $\Lambda = \{\lambda, \gamma, \mu, \sigma, \pi, \theta, \alpha, \beta, \delta, \epsilon, \kappa, \rho, \nu, \tau\}$.

Creative/Brainstorm modes: Apply steps 3–4 only.

Deduplication: Remove exact string duplicates across all samples, yielding K' unique samples where $K' \leq K$.

A.3.3 *Centroid and Modal Sample Computation.*

Let $\{s_1, s_2, \ldots, s_{K'}\}$ denote unique samples and $\{\mathbf{e}_1, \mathbf{e}_2, \ldots, \mathbf{e}_{K'}\}$ their L2-normalized embeddings.

Centroid:

$$\mathbf{c} = \frac{\sum_{i=1}^{K'} \mathbf{e}_i}{\left\|\sum_{i=1}^{K'} \mathbf{e}_i\right\|} \tag{3}$$

Modal sample:

$$i^* =_{i \in \{1,\ldots,K'\}} d(\mathbf{e}_i, \mathbf{c}) =_i (1 - \mathbf{e}_i^\top \mathbf{c}) \tag{4}$$

Sample s_{i^*} is the modal (most consensus-representative) response.

A.4 Negative Concept Extraction

A.4.1 *Creative Mode: YAKE-Based Consensus Phrases.*

Step 1: Candidate extraction

Apply YAKE keyword extractor with $n_{\max} = 4$ (maximum n-gram length) and $k_{\text{top}} = 120$ (candidate pool size) to concatenated sample text. YAKE returns tuples $(phrase, score)$ where lower score indicates higher keyword quality.

Step 2: Phrase-only filtering

Require ≥ 2 alphabetic words per phrase:

$$\text{keep}(phrase) \iff |\{w : w \in \text{words}(phrase), w \text{ is alphabetic}\}| \geq 2 \tag{5}$$

Step 3: Generic content filtering

Define stopword set $\mathcal{S}$ (ENGLISH_STOP_WORDS) and generic content set $\mathcal{G}$:

> $\mathcal{G}$ = {time, people, person, thing, things, stuff, way, ways, life, lives, world, good, bad, really, very, just, like, also, still, often, sometimes, always, never, many, much, make, made, making, use, used, using, get, got, getting, go, going, say, says}

Content words: $C(phrase) = \{w : w \in \text{words}(phrase), w \notin \mathcal{S}\}$

Discard if $|C(phrase)| < 2$ or $|C(phrase) \cap \mathcal{G}| \geq |C(phrase)|/2$.

Step 4: Cross-sample consensus

Define frequency: $f(phrase) = |\{s_i : \text{normalized}(phrase) \in \text{normalized}(s_i)\}|$

Keep only phrases with $f(phrase) \geq 2$.

Step 5: Prompt exclusion

Let W_p = set of words in prompt p. Discard if $\text{words}(phrase) \subseteq W_p$.

Step 6: Ranking

Sort by $(f(phrase), -score)$ descending; select top 12.

A.4.2 *Technical Mode: Curated Style Boilerplate.*

Maintain fixed list $\mathcal{F}_{\text{tech}}$ of style phrases:

> {"let's break down", "let's delve into", "let's dive into", "here are the differences", "let's compare them", "in this section", "in summary", "to summarize", "overall,", "overall this means", "in conclusion", "as you can see"}

Return only phrases from $\mathcal{F}_{\text{tech}}$ observed in ≥ 1 sample. Maximum 12 concepts.

A.4.3 *Brainstorm Mode: Frame Detection + Fallback.*

Curated frame list $\mathcal{F}_{\text{brainstorm}}$:

{"revolutionary solution", "game changer", "cutting edge", "next level", "unlock the potential", "transform the way", "in today's world", "think outside the box", "at the end of the day", "moving forward", "low hanging fruit", "one size fits all", "quick win"}

Add observed frames from $\mathcal{F}_{\text{brainstorm}}$ to concept list. If $|concepts| < 12$, apply Creative mode YAKE extraction to fill remainder.

A.5 Semantic Radar Visualization

A.5.1 Dimensionality Reduction. Construct matrix $\mathbf{E} \in \mathbb{R}^{(K'+1)\times d}$ containing sample embeddings and centroid. If user draft provided, append its embedding: $\mathbf{E} \in \mathbb{R}^{(K'+2)\times d}$.

UMAP projection (if $K' \geq 6$):

```
reducer = umap.UMAP(
    n_neighbors=min(10, K'-1),
    min_dist=0.15,
    metric='cosine',
    random_state=42
)
X_2d = reducer.fit_transform(E)
```

PCA fallback (if $K' < 6$):

```
reducer = PCA(n_components=2, random_state=42)
X_2d = reducer.fit_transform(E)
```

A.5.2 Kernel Density Estimation. Let $\mathbf{X}_{\text{samples}} \in \mathbb{R}^{K'\times 2}$ denote projected sample coordinates (excluding centroid and user draft).

Gaussian KDE:

$$\hat{f}(\mathbf{x}) = \frac{1}{K'} \sum_{i=1}^{K'} \mathcal{N}(\mathbf{x}; \mathbf{X}_{\text{samples}}[i], \Sigma) \tag{6}$$

where Σ is estimated via Scott's rule.

Evaluate $\hat{f}$ on 120×120 grid spanning data range ±0.8 margin. Render as contour plot with opacity 0.35.

A.5.3 Plot Elements.

- **Sample points:** Scatter plot with hover text showing full sample content. Optional numeric labels (0, 1, ..., $K' - 1$) if detail mode enabled.
- **Centroid:** Distinct marker with label "centroid" and hover text showing modal sample.
- **User draft:** (Optional) Distinct marker with label "you" if draft provided.

A.6 Repulsion Engine Algorithm

A.6.1 Contrastive Decoding with Targeted Penalties. **Input:** Prompt p, mode $\mathcal{M}$, negative concepts $\mathcal{N}$, hyperparameters λ (repulsion), τ (fluency floor), β (phrase penalty), ρ (repetition penalty), n_{ngram} (no-repeat window), τ_{temp} (temperature), p_{nucleus} (top-p), L_{gen} (max generation tokens), seed.

Preprocessing:

(1) Format input: $x = \text{apply\_chat\_template}(p, \text{system} = \text{get\_mode\_prompt}(\mathcal{M}))$

(2) Tokenize: $\mathbf{x}_0 = \text{tokenize}(x)$

(3) Build negative token set: $\mathcal{T}_{\text{neg}} = \bigcup_{phrase \in \mathcal{N}} \text{tokenize}(\text{" "} + phrase)$

(4) Prime KV caches:

```
out_s = strong_model(x_0, use_cache=True)
out_w = weak_model(x_0, use_cache=True)
cache_s, cache_w = out_s.past_key_values, out_w.past_key_values
```

(5) Initialize: $y = []$ (generated tokens), n-gram map $\mathcal{M}_{\text{ngram}}$ from $\mathbf{x}_0$

Generation loop ($t = 1$ to L_{gen}):

Step 1: Compute base logits

$$\ell_s^{(t)} = \text{strong\_model}(y_{t-1}, \text{past} = \text{cache}_s).logits[:, -1, :] \in \mathbb{R}^{|V_s|} \tag{7}$$

$$\ell_w^{(t)} = \text{weak\_model}(y_{t-1}, \text{past} = \text{cache}_w).logits[:, -1, :] \in \mathbb{R}^{|V_w|} \tag{8}$$

Step 2: Align weak logits

$$\ell_{\text{w,aligned}}^{(t)}[i] = \begin{cases} \ell_{\text{w}}^{(t)}[m_i] & \text{if } m_i \geq 0 \\ 0 & \text{otherwise} \end{cases} \tag{9}$$

Step 3: Contrastive combination

$$\ell_{\text{contrast}}^{(t)} = \ell_s^{(t)} - \lambda \cdot \ell_{\text{w,aligned}}^{(t)} \tag{10}$$

Step 4: Fluency floor mask

$$\ell_{\text{contrast}}^{(t)}[i] \leftarrow -\infty \quad \text{if} \quad \frac{\exp(\ell_s^{(t)}[i])}{\sum_j \exp(\ell_s^{(t)}[j])} < \tau \tag{11}$$

Step 5: Block disallowed characters

Build set $\mathcal{T}_{\text{disallowed}}$ of token IDs containing non-ASCII alphabetic characters (excluding Λ):

$$\ell_{\text{contrast}}^{(t)}[i] \leftarrow -\infty \quad \forall i \in \mathcal{T}_{\text{disallowed}} \tag{12}$$

Step 6: Phrase-derived token penalty

$$\ell_{\text{contrast}}^{(t)}[i] \leftarrow \ell_{\text{contrast}}^{(t)}[i] - \beta \quad \forall i \in \mathcal{T}_{\text{neg}} \tag{13}$$

Step 7: No-repeat n-gram blocking

Let $\text{prefix}_{n-1} = (y_{t-n+1}, \ldots, y_{t-1})$ be last $n-1$ tokens.

If $\text{prefix}_{n-1} \in \mathcal{M}_{\text{ngram}}$:

$$\ell_{\text{contrast}}^{(t)}[i] \leftarrow -\infty \quad \forall i \in \mathcal{M}_{\text{ngram}}[\text{prefix}_{n-1}] \tag{14}$$

Step 8: Repetition penalty

For each i such that $i \in y$ (already generated):

$$\ell_{\text{contrast}}^{(t)}[i] \leftarrow \begin{cases} \ell_{\text{contrast}}^{(t)}[i]/\rho & \text{if } \ell_{\text{contrast}}^{(t)}[i] > 0 \\ \ell_{\text{contrast}}^{(t)}[i] \cdot \rho & \text{otherwise} \end{cases} \tag{15}$$

Step 9: Temperature scaling

$$\ell_{\text{contrast}}^{(t)} \leftarrow \ell_{\text{contrast}}^{(t)}/\max(\tau_{\text{temp}}, 10^{-6}) \tag{16}$$

Step 10: Top-p filtering
Sort logits descending: $(\ell_{\text{sorted}}, \text{idx}_{\text{sorted}})$
Compute cumulative softmax probabilities: $q_i = \sum_{j=1}^{i} \frac{\exp(\ell_{\text{sorted}}[j])}{\sum_k \exp(\ell_{\text{sorted}}[k])}$
Mask: $\ell_{\text{sorted}}[i] \leftarrow -\infty$ if $q_i > p_{\text{nucleus}}$ and $i > 1$
Scatter back: $\ell^{(t)}_{\text{contrast}} \leftarrow \text{scatter}(\ell_{\text{sorted}}, \text{idx}_{\text{sorted}})$
Step 11: Sample

$$p^{(t)} = \text{softmax}(\ell^{(t)}_{\text{contrast}}) \tag{17}$$

If $\text{isnan}(p^{(t)})$ or $\sum_i p_i^{(t)} = 0$, fall back to strong-only sampling with same constraints.
Otherwise: $y_t \sim \text{Categorical}(p^{(t)})$
Step 12: Update state

- Append y_t to y
- Update n-gram map: $\mathcal{M}_{\text{ngram}}[\text{prefix}_{n-1}] \leftarrow \mathcal{M}_{\text{ngram}}[\text{prefix}_{n-1}] \cup \{y_t\}$
- Update KV caches via incremental forward pass

Step 13: Check termination
If $y_t = \text{eos\_token\_id}$ or $t = L_{\text{gen}}$, break.
Post-processing:

(1) Decode: output = $\text{detokenize}(y)$
(2) Word-level cleanup: Remove words with non-ASCII alphabetic chars (excluding Λ)
(3) Technical mode only: Apply filler removal, deduplication, formula normalization

A.6.2 Fallback Sampling. If softmax produces NaN or zero sum:

```
logits_fallback = strong_logits.clone()
logits_fallback[:, disallowed_ids] = -inf
logits_fallback[:, ngram_banned_ids] = -inf
logits_fallback = logits_fallback / temperature
logits_fallback = top_p_filter(logits_fallback, p_nucleus)
probs = softmax(logits_fallback)
sample from probs
```

A.7 Optional Polish Pass (Creative Mode Only)

System prompt:

> *You are an editor. Fix spelling, grammar, and broken sentences ONLY. Do NOT add new imagery or new content. Do NOT change meaning. Do NOT reintroduce banned phrases. [If dialogue constraint detected:] Ensure the final line is exactly one line of dialogue in quotes. Avoid these phrases: [list negative concepts].*

User prompt:

> *Text to minimally polish:\n\n[repulsed_output]\n\nReturn ONLY the corrected text.*

Generation: Greedy decoding (do_sample=False), max 240 tokens.

Dialogue constraint detected if prompt or extra instructions contain: "single line of dialogue", "end with a single line of dialogue", or "end with dialogue".

A.8 Default Hyperparameters by Mode

Parameter	Creative	Technical	Brainstorm
λ (contrastive repulsion)	1.2	0.6	1.5
τ (fluency floor)	0.003	0.010	0.002
β (phrase penalty)	4.0	4.0	4.0
ρ (repetition penalty)	1.06	1.12	1.06
n_{ngram} (no-repeat window)	4	6	4
τ_{temp} (temperature)	1.0	N/A	1.0
p_{nucleus} (top-p)	0.9	N/A	0.9
K (look-ahead samples)	12	12	12
$L_{\text{lookahead}}$ (look-ahead tokens)	128	128	128
L_{gen} (generation tokens)	320	220	260
Polish pass	Optional	No	No
Baseline decoding	Sampling	Greedy	Sampling

Table 2. Mode-specific default hyperparameters. Technical mode uses greedy decoding with repetition constraints for baseline; Creative and Brainstorm use nucleus sampling.

A.9 Evaluation Metrics

A.9.1 Distance from Mode. For generated text g, compute embedding $\mathbf{e}_g$ and cosine distance from centroid:

$$D_{\text{mode}}(g) = 1 - \mathbf{e}_g^\top \mathbf{c} \tag{18}$$

Compute separately for:

- Baseline output: D_{baseline}
- Repulsed output: D_{repulsed}
- User draft (optional): D_{draft}

Higher distance indicates greater divergence from consensus.

A.9.2 Reported Metrics. Interface displays JSON object:

```
{
  "mode": mode_name,
  "k_used": K_prime,
  "baseline_distance_from_mode": D_baseline,
  "repulsed_distance_from_mode": D_repulsed,
  "your_draft_distance_from_mode": D_draft (if provided),
  "seconds": runtime,
  "models": {"strong": model_name_s, "weak": model_name_w},
  "vocab_overlap_ratio": overlap_ratio,
  "polish": polish_enabled (Creative only)
}
```