OOCO: Objective-Oriented Content Optimization on Discrete Symbolic Manifolds

Anonymous ACL submission

Abstract

Traditional text rewriting frameworks rely on intrinsic surface-level metrics like fluency, failing to establish a link between text generation and explicit reader impact. This paper proposes Objective-Oriented Content Optimization (OOCO), a framework that formalizes content rewriting as a constrained stochastic optimization problem over discrete symbolic manifolds to optimize measurable behavioral reward signals. To overcome the challenge of discrete tokens, which prevents direct numerical optimization, we propose two solutions: Gradient-to-Strategy Mapping (GSM), which decodes localized directional derivatives into rewriting instructions for a Large Language Model agent, and Gradient-Guided Empirical Search (GGES), a multi-phase framework that integrates continuous latent space scouting with discrete space pruning. Empirical evaluations across two tasks—generalized digital virality and hyper-personalized click-through optimization—demonstrate that OOCO significantly outperforms random search and zero-shot baselines. On the virality task, our optimal configuration achieved a 22.10% mean reward improvement, representing a 4.3× gain in reward. On the personalization task, our approaches achieved a mean reward improvement of up to 88.2%, exceeding the best baseline by an absolute gain of nearly 28 percentage points. Cross-architecture validation confirms that OOCO produces durable, structurally stable linguistic transformations that preserve core semantic fidelity.

1 Introduction

The evolution of Natural Language Generation (NLG) has historically followed a trajectory defined by the pursuit of formal linguistic excellence, moving from early template-based approaches to the current dominance of Large Language Models (LLM) (Gatt and Krahmer, 2018). Throughout this progression, the primary metrics of success have remained centered on fluency, coherence, and stylistic alignment with human-authored corpora (Novikova et al., 2017). While these advancements have yielded models capable of producing text that is increasingly indistinguishable from human writing, they operate without an explicit, quantifiable link to the external impact that text exerts on a reader (Hashimoto et al., 2019). This paper argues that the current focus on surface-level properties represents a plateau in NLG research. We propose a shift toward Objective-Oriented Content Optimization (OOCO), a framework wherein content rewriting is treated as a constrained stochastic optimization problem designed to maximize[1] specific, measurable reader reactions.

In this paradigm, the quality of a text is no longer an intrinsic property judged by n-gram overlap metrics such as BLEU (Papineni et al., 2002) or ROUGE (Lin, 2004). Such metrics often fail to correlate with real-world task effectiveness or behavioral change (Reiter and Belz, 2009). Instead, we define quality as the expected value of an objective function or reward signal R over a distribution of human behavioral responses D. This approach addresses a fundamental gap in communicative AI: the transition from conveying information to achieving an objective. Whether the goal is to drive engagement, enhance educational retention, or facilitate faster information processing, the linguistic choices must be optimized as variables within a high-dimensional strategy space.

The transition to a mathematical objective function offers a distinct advantage over heuristic-based editing. Traditional rewriting often relies on generalized linguistic best practices which serve as noisy proxies for actual objectives. Framing the task as a mathematical optimization allows for a systematic traversal of the rewriting strategy space to identify

[1] Or minimize, depending on the objective function. For simplicity, we frame all tasks as maximization throughout this paper.

a global optimum for a given context, rather than relying on the localized, often biased, stylistic preferences of an editor. This formulation also enables true Hyper-Personalization by conditioning the reward signal on specific audience characteristics. In this context, the objective is to maximize the conditional expected reward, $E[R \mid x, u]$, where u represents a vector of user-specific features. An OOCO system customizes text to target the unique psychological traits of specific individuals or demographics. It preserves the message's original meaning while adjusting its delivery style to maximize behavioral impact.

Because text is inherently discrete, standard continuous optimization techniques like gradient-based backpropagation cannot be directly applied to the output token space to optimize a mathematical objective function. Furthermore, text quality is highly sensitive to token permutations, requiring precise sequence ordering to remain readable, and the optimization must adhere to strict semantic invariance, preserving the underlying concepts of the source text without introducing fabricated information. We address this by two iterative rewriting approaches that use an LLM as a rewriting agent and treat the gradient of an objective function as rewriting instructions. The textual gradient paradigm is inspired by the the work of Zhou et al. (2023). We investigate two distinct mechanisms for mapping gradients to rewriting instructions: 1) Cosine similarity to align the gradient vector with a predefined embedding space of rewriting strategies; 2) A specialized "Interpreter Model", a multi-class classifier to decode gradients into discrete rewriting strategies (inspired by Probing and Interpretation (Belinkov, 2022))

Finally, we empirically evaluate our proposed approaches across two distinct optimization tasks: generalized content optimization for digital virality maximization and hyper-personalized marketing email rewriting (to maximize click-through probability). The experimental results demonstrate that our framework effectively solves both problems, outperforming the baselines to varying degrees across the evaluation metrics. Specifically, the proposed approaches yield substantial gains in the primary objective function while maintaining strict adherence to the natural language and semantic constraints.

2 Related Work

Conceptually, this work is situated at the intersection of Controllable Text Generation (CTG) and Behavioral Economics. Traditional CTG has focused on hard constraints (e.g., ensuring a specific keyword appears) or soft stylistic constraints (e.g., sentiment or politeness). Ficler and Goldberg (2017) established early frameworks for controlling linguistic attributes via latent variable modeling and conditioned generators. However, these methods remain internal to the model's latent space; they do not explicitly account for the external, often non-linear, reaction of the reader.

The conceptual move from style to objective mirrors the development of Persuasion Architecture in social psychology. Cialdini (2001) identified specific rhetorical levers—such as authority, scarcity, and social proof—that predictably shift human behavior. Recent efforts in Computational Persuasion have attempted to automate these levers (Durmus and Cardie, 2018), but they often rely on static templates or categorical classifiers. Our framework extends this by treating the optimization as continuous and data-driven, leveraging real-world behavioral data rather than fixed psychological heuristics.

Content personalization has undergone a fundamental transition from retrieval-based recommendation toward generative optimization (Salemi et al., 2024). While some methodologies utilize Direct Preference Optimization (DPO) and Reinforcement Learning from Human Feedback (RLHF) to align model outputs with latent user preferences (Rafailov et al., 2024; Ouyang et al., 2022), they frequently lack the deterministic control required for objective-oriented text revision under strict semantic constraints. Content hyper-personalization extends personalization by adapting text to individual interaction histories rather than broad demographic groups. Modern frameworks typically execute this individual adaptation through three technical paradigms: in-context learning over retrieval-augmented user histories (Salemi et al., 2024; Wu et al., 2023), parameter-efficient fine-tuning via user-specific modular adapters (Zhang et al., 2024), or latent activation steering to shift hidden model states at inference time (Liu et al., 2025). While these methodologies effectively align text with past user behaviors, they focus primarily on stylistic replication. They do not formalize text generation as an explicit, objective-driven optimization problem designed to optimize a specific external metric.

Our framework differentiates itself from these approaches by directly optimizing the text against a mathematically defined objective function, while treating linguistic coherence and semantic alignment as mathematical constraints rather than passive side-effects stylistic rewriting.

3 Methodology

The core of our framework, OOCO, formulates the task of text revision as a high-dimensional, constrained stochastic optimization problem, where the objective is the maximization of an external behavioral reward signal.

3.1 Problem Formulation and Challenges

Let $\mathcal{X}$ be the discrete space of all possible natural language sequences. Given an initial source sequence $x_0 \in \mathcal{X}$, our objective is to find an optimal transformed sequence x^* that maximizes a scalar reward R. In our framework, R is defined as the expectation over a distribution of human behavioral responses D. We formulate this as a constrained optimization problem:

$$\begin{aligned} x^* &= \arg\max_{x \in \mathcal{X}} \mathbb{E}_{y \sim P(D|x,u)}[f(y)] \\ &\text{subject to: } \mathcal{D}_{sim}(x, x_0) \leq \epsilon \text{ and } \mathcal{P}(x) \leq \tau \end{aligned} \tag{1}$$

Where $u \in \mathbb{R}^k$ represents a vector of audience or context-specific features, serving as the conditioning variable for hyper-personalization; y is a stochastic variable representing the reader's reaction, modeled as a distribution to account for inherent human variance; and $f(y)$ is a utility function mapping behavioral responses to a scalar reward. The reward surface is approximated via a surrogate neural function $\hat{R}_\phi(x, u) \approx \mathbb{E}[f(y) \mid x, u]$, typically implemented as a predictive classifier (e.g., estimating the probability of a binary engagement event). While $\mathcal{D}_{sim}$ and $\mathcal{P}$ represent formal boundaries for semantic fidelity and linguistic fluency, respectively, these constraints are addressed through textual instructions to a rewriting agent, and a post rewriting LLM-as-a-Judge for semantic fidelity.

Solving this optimization problem is not straightforward. The primary challenge arises from the discrete combinatorial nature of the sequence space $\mathcal{X}$, which renders traditional gradient-based optimization—the standard for navigating continuous Euclidean spaces—largely inapplicable. Given a vocabulary V and a sequence of length n, the search space expands at a rate of $|V|^n$, creating a disjoint token manifold where local directional derivatives cannot be computed via standard autodifferentiation. The textual domain presents a discrete, non-differentiable optimization landscape. Consequently, localized perturbations—such as single-token substitutions—can induce phase shifts that compromise semantic fidelity and linguistic fluency.

3.2 Proposed Approaches

We propose two distinct approaches to solve the optimization problem defined in Section 3.1 (Figure 1). Each must navigate the core trade-off between maximizing objective reward and adhering to the constraints (maintaining text integrity). While aggressive modifications risk introducing semantic drift or hallucinations, conservative edits fail to optimize the objective. These strategies allow us to contrast two distinct optimization paradigms: one prioritizing iterative local stability, and the other favoring broader exploration.

To operationalize these algorithms, we have constructed a high-dimensional rewriting strategy library $\mathcal{S}$ (see Appendix A), which will be used as instruction to a rewriting LLM, based on five fundamental rhetorical categories: The Vibe & Tone, Clarity & Precision, Structure & Flow, Complexity & Depth, and Innovation & Execution. Each discrete rewriting strategy is defined by a unique combination of one to five adjectives, with only one descriptor selected from each category. Given that each of the five categories contains ten distinct adjectives, this combinatorial approach generates a vast space of 11^5 potential instructional permutations. While this granularity approximates continuous gradient shifts, the resulting high cardinality of the strategy space requires efficient selection and pruning mechanisms.

In the rewriting optimization problem (Equation 1), two distinct language models are involved: the pre-trained encoder model that converts the text into the input of the objective function, and the rewriting LLM agent. OOCO avoids any mathematical conflicts between these two systems by keeping the gradient optimization strictly inside the encoder's embedding space. The chosen strategies are then given to the rewriting LLM as plain text instructions, meaning the two models never need to share or align their internal vector representations.

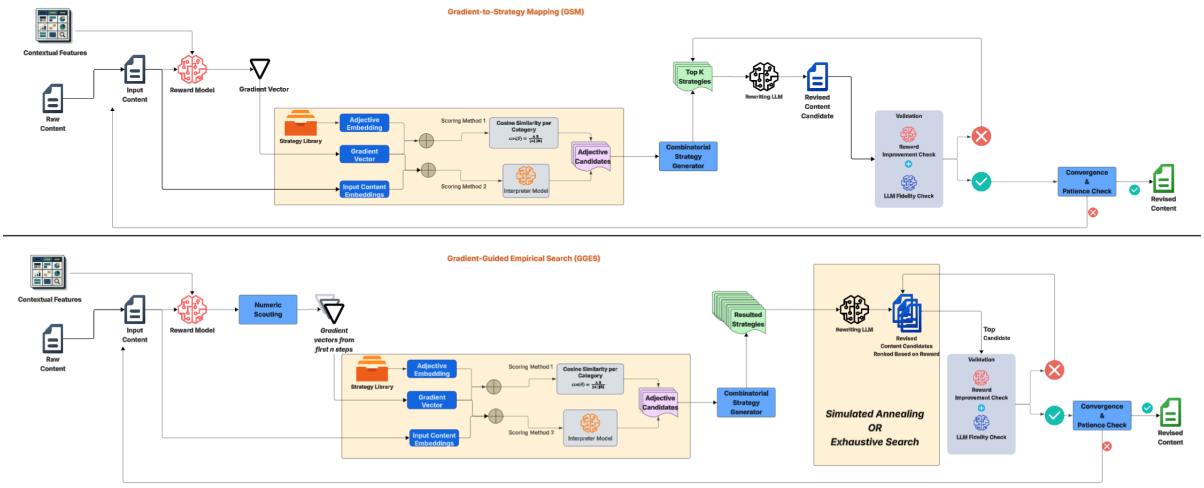

Figure 1: The Objective-Oriented Content Optimization (OOCO) architecture.

3.2.1 Approach 1: Gradient-to-Strategy Mapping (GSM)

To bridge the gap between continuous optimization and discrete text, this iterative approach converts numerical gradients into natural language instructions. Instead of attempting to optimize the text directly, we compute the gradient of the objective function with respect to the input embeddings, $\nabla_{\mathbf{z}} \hat{R}_\phi$. This gradient represents the directional change required to maximize the expected reward at a given iteration. We then map this vector to a discrete rhetorical strategy s from the rewriting strategy library $\mathcal{S}$, which guides the rewriting LLM. Each strategy is represented as a vector $\mathbf{e}_s$ in the same embedding space as the gradient. We investigate two mechanisms for this mapping: the first mechanism selects the strategy that has the highest cosine similarity to the reward gradient:

$$s^* = \arg\max_{s \in \mathcal{S}} \text{cos-sim}(\nabla_{\mathbf{z}} \hat{R}_\phi, \mathbf{e}_s) \tag{2}$$

This selection ensures that the instruction given to the rewriting model aligns with the direction of the greatest predicted improvement in the reward signal. In the second mechanism, we use a trained interpreter model (see Appendix B) to translate the gradient. This model is a multi-class classifier with rewriting strategies from the rewriting strategy library $\mathcal{S}$ as its target variable, and gradients of a large-scale, highly diverse parallel corpus of original and stylistically altered texts as the input features.

Algorithm 1 Gradient-to-Strategy Mapping (GSM)

Input: Source text x_0, Audience features u, Reward model $\hat{R}_\phi$, Max iterations T, Patience limit P, Strategy limit k, Improvement threshold δ

Output: Optimized text x^*

Initialize $x^{(0)} = x_0$, $x^* = x_0$, $R^* = -\infty$, patience $= 0$.

For $t = 0$ to $T - 1$:

- **a.Extract Latent Representation:** Compute embedding $\mathbf{z}^{(t)}$ for $x^{(t)}$.
- **b.Compute Gradient:** Calculate $\mathbf{g}^{(t)} = \nabla_{\mathbf{z}} \hat{R}_\phi(\mathbf{z}^{(t)}, u)$.
- **c.Strategy Search:** Identify the top k strategies in library $\mathcal{S}$ via:
  - **•Option 1 (Cosine):** Rank by cosine similarity $(\mathbf{g}^{(t)}, \mathbf{e}_s)$.
  - **•Option 2 (Interpreter):** Pass $\mathbf{z}^{(t)}$, $\mathbf{g}^{(t)}$ through interpreter model.
- **d.For each** strategy s_i in the top k ranked strategies:
  - **i.Rewrite:** Generate candidate x' using LLM with $x^{(t)}$ and s_i.
  - **ii.Improvement Check: If** $\hat{R}_\phi(x') - \hat{R}_\phi(x^{(t)}) \geq \delta$:
    - **Verify:** Apply LLM-as-a-Judge for semantic fidelity (x_0, x').
    - **If** valid: Set $x^{(t+1)} = x'$ and **Break** to iteration $t+1$.
    - **Else:** Continue to next best strategy s_{i+1}.
  - **Else:** Continue to next best strategy s_{i+1}.
- **e.Termination Check: If** ranked set exhausted without a valid, improving candidate, set $x^{(t+1)} = x^{(t)}$ and patience $=$ patience $+ 1$. **Else if** $\hat{R}_\phi(x^{(t+1)}) > R^*$: set $x^* = x^{(t+1)}$, $R^* = \hat{R}_\phi(x^{(t+1)})$, patience $= 0$. **Else:** patience $=$ patience $+ 1$. **If** patience $\geq P$: **Break** main loop.

Return: x^*

3.2.2 Approach 2: Gradient-Guided Empirical Search (GGES)

While Approach 1 focuses on localized, sequential refinement, Gradient-Guided Empirical Search (GGES) utilizes a multi-phase search process designed to evaluate a wider breadth of rhetorical transformations. This approach is executed in two distinct phases per iteration. The process begins with a Numerical Scouting phase, where the frame-

work executes a series of gradient ascent steps in the continuous latent space starting from the text embedding $\mathbf{z}^{(0)}$. This identifies a theoretical trajectory toward an optimum, unconstrained by the requirements of natural language. In the subsequent Strategic Pruning phase, all gradients collected during scouting are converted into strategies (similar to Approach 1 either through cosine similarity or the interpreter model) containing at least one aligned adjective. The framework then generates a comprehensive set of candidate strategies based on all combinations of the adjectives identified. If the resulting number of permutations is smaller than a threshold M, the system performs an exhaustive rewrite across all candidates; otherwise, a search algorithm, such as Simulated Annealing (SA), is employed to identify the strategy that maximizes the realized objective function, efficiently. Following the selection of a candidate, the resulting output is audited by the LLM-as-a-Judge for semantic fidelity. If verified, the final candidate x^* of that iteration serves as the new input x_0 for a subsequent cycle of numerical scouting and empirical search. This macro-recursive loop continues until a convergence criteria is met or a maximum number of iterations is reached.

3.3 LLM Prompts and Optimization Constrains

Both of the proposed approaches rely on a set of LLM calls for strategy-guided rewriting, and post rewriting semantic fidelity check. Optimization constraints introduced in 3.1, namely $\mathcal{D}_{\text{sim}}(x, x_0) \leq \epsilon$ and $\mathcal{P}(x) \leq \tau$ are directly injected into the rewriting prompt as textual instructions and validated in the post rewriting LLM-as-a-Judge step. These prompts are introduced in Appendix C.

4 Experiments

We designed and conducted two experiments using two distinct datasets. These experiments were designed to demonstrate the framework's adaptability to both generalized improvement towards an objective, and hyper-personalization.

4.1 Experiment Design

The first experiment utilizes the publicly available Mashable News dataset, where the reward model is trained to predict the virality (a binary measure of digital sharing) of articles, testing the framework's ability to optimize text for broad audience appeal (for details on the data, the objective function, and model training, see Appendix D). Two hundred news articles are randomly selected as the final rewriting dataset.

The second experiment uses a proprietary dataset of marketing emails, with the objective of hyper-personalized rewriting to maximize user engagement (click-through) probability (for details on the data, the objective function, and model training, see Appendix E). Five hundred marketing emails are randomly selected as the final rewriting dataset for this experiment.

Algorithm 2 Gradient-Guided Empirical Search (GGES)

Input: Source text x_0, Audience features u, Reward model $\hat{R}_\phi$, Threshold M, Improvement threshold δ, Max iterations T, Patience limit P

Output: Optimized text x^*

Initialize $x^{(0)} = x_0$, $x^* = x_0$, $R^* = \hat{R}_\phi(x_0, u)$, patience $= 0$.

For $t = 0$ to $T - 1$:

a.Numerical Scouting:
- Calculate initial embedding $\mathbf{z}^{(0)}$ for $x^{(t)}$.
- Perform gradient ascent in latent space for n steps: $\mathbf{z}^{(i+1)} = \mathbf{z}^{(i)} + \eta \nabla_{\mathbf{z}} \hat{R}_\phi(\mathbf{z}^{(i)}, u)$
- Extract and store the sequence of gradients $\{\mathbf{g}^{(1)}, \ldots, \mathbf{g}^{(n)}\}$ along the trajectory.

b.Strategic Pruning:
- For each $\mathbf{g}^{(i)}$, identify adjectives from the strategy library $\mathcal{S}$ via either cosine similarity (Option 1) or interpreter model (Option 2).
- Construct candidate set $\mathcal{S}_{cand}$ from all adjective combinations.

c.Empirical Selection & Verification: Rank $\mathcal{S}_{cand}$ by predicted reward via LLM rewriting (exhaustive search if $\leq M$, else Simulated Annealing). **For each** candidate x' in ranked set:

i.Verify: Apply LLM-as-a-Judge for semantic fidelity.

ii.If x' is valid and $\hat{R}_\phi(x') - \hat{R}_\phi(x^{(t)}) \geq \delta$: Set $x^{(t+1)} = x'$; **Break** to Termination Check.

iii.Else if x' is valid but no improvement: Set $x^{(t+1)} = x^{(t)}$; **Break** to Termination Check.

iv.Else: Continue to next best candidate in ranked set.

d.Termination Check: **If** ranked set exhausted without a valid, improving candidate, set $x^{(t+1)} = x^{(t)}$ and patience $=$ patience $+ 1$. **Else if** $\hat{R}_\phi(x^{(t+1)}) > R^*$: set $x^* = x^{(t+1)}$, $R^* = \hat{R}_\phi(x^{(t+1)})$, patience $= 0$. **Else:** patience $=$ patience $+ 1$. **If** patience $\geq P$: **Break** main loop.

Return: x^*

4.2 Evaluation

4.2.1 Baseline Benchmarks

To evaluate the performance of OOCO framework, we establish two LLM-driven baselines. Rather than comparing against naive, unprompted rewrites, both baselines are heavily context-conditioned to leverage the LLM's intrinsic reasoning capabilities.

Category	Metric	Description	Goal
Reward Opt.	Mean Imp. (%)	Average relative increase in scalar reward $\hat{R}_\phi$ across the corpus.	Maximize
	Improved (%)	Proportion of rewritten texts where $\hat{R}_\phi(x^*) > \hat{R}_\phi(x_0)$.	Maximize
Constraints	Semantic BES	Bidirectional entailment score to ensure semantic fidelity and prevent hallucination.	Maximize
	Length Ratio	Symmetric ratio $\min(\lvert x^*\rvert/\lvert x_0\rvert, \lvert x_0\rvert/\lvert x^*\rvert)$ to ensure structural adherence.	≈ 1.0
	Perplexity	Cross-entropy of the text under a reference LM to assess linguistic fluency.	Minimize
	Coherence	Mean cosine similarity between adjacent sentence embeddings for logical flow.	Maximize

Table 1: Evaluation metrics for OOCO performance. The metrics are divided into reward-driven gains (Optimization) and quality-preservation standards (Constraints).

		Reward Objective		**Independent Grader**		**Constraint Satisfaction Metrics**			
Method	**Model**	**Mean Δ**	**% Imp.**	**Mean Δ**	**% Imp.**	**BES ↑**	**Len. ≈ 1**	**PPL ↓**	**Coh. ↑**
Experiment I: Mashable Content Optimization (Virality)									
Baseline (ZSCCP)	Llama-3.3	3.0%	43.5%	0.3%	50.0%	0.854	0.924	0.472	0.384
Baseline (ZSCCP)	gpt-oss-120b	3.5%	51.5%	0.2%	24.0%	**0.987**	**0.950**	0.441	0.321
Baseline (BNRS)	Llama-3.3	3.1%	43.5%	0.3%	**56.0%**	0.807	0.864	0.455	0.364
Baseline (BNRS)	gpt-oss-120b	5.2%	54.5%	0.2%	26.0%	0.981	0.922	0.448	0.314
GSM-Cosine	Llama-3.3	7.9%	65.5%	2.8%	35.5%	0.835	0.924	0.472	0.384
GSM-Cosine	gpt-oss-120b	18.5%	85.0%	3.7%	40.0%	0.647	0.594	0.444	0.362
GSM-Interpreter	Llama-3.3	7.1%	65.5%	3.2%	37.0%	0.806	0.936	0.480	0.386
GSM-Interpreter	gpt-oss-120b	19.8%	85.0%	**4.8%**	48.5%	0.594	0.585	0.446	**0.403**
GGES	Llama-3.3	7.6%	67.9%	3.1%	38.4%	0.845	0.914	0.466	0.375
GGES	gpt-oss-120b	**22.1%**	**91.3%**	4.0%	48.0%	0.654	0.558	**0.430**	0.374
Experiment II: Marketing Email Hyper-Personalization (Click-Through)									
Baseline (ZSCCP)	Llama-3.3	18.6%	79.0%	17.4%	70.6%	0.833	0.854	0.452	**0.437**
Baseline (ZSCCP)	gpt-oss-120b	12.6%	74.0%	10.9%	61.4%	0.825	0.883	0.460	0.375
Baseline (BNRS)	Llama-3.3	59.1%	**97.0%**	**39.9%**	**85.6%**	0.939	0.763	0.469	0.354
Baseline (BNRS)	gpt-oss-120b	60.5%	94.0%	32.9%	75.0%	0.918	0.894	0.499	0.340
GSM-Cosine	Llama-3.3	49.0%	92.0%	23.7%	69.2%	0.925	0.828	0.486	0.402
GSM-Cosine	gpt-oss-120b	81.9%	90.2%	33.9%	68.0%	0.883	0.907	0.486	0.361
GSM-Interpreter	Llama-3.3	49.0%	93.0%	32.1%	80.0%	0.945	0.819	0.455	0.401
GSM-Interpreter	gpt-oss-120b	**88.2%**	94.0%	38.5%	76.4%	0.877	**0.928**	**0.433**	0.348
GGES	Llama-3.3	29.1%	90.4%	19.0%	67.0%	**0.951**	0.852	0.499	0.401
GGES	gpt-oss-120b	59.5%	76.4%	29.6%	59.0%	0.884	0.848	0.534	0.380

Table 2: Full-scale experimental results evaluating the OOCO frameworks. Content strategies are explicitly mapping-restricted to preserve space constraints. Metrical scores (BES, Length, PPL, and Coherence) are uniformly scaled.

Zero-Shot Context-Conditioned Prompting (ZSCCP) Our first baseline evaluates the LLM's native ability to optimize text, without the aid of gradient-derived stylistic strategies. In case of hyper-personalization for each item in the evaluation set, we dynamically construct a prompt that injects the personalized tabular features directly into the context window. To help the LLM interpret the numerical metadata, continuous features are framed relative to the broader population distribution (e.g., presenting a value alongside the population mean and standard deviation). The LLM is instructed to rewrite the source text to maximize engagement for this specific target profile while strictly adhering to semantic fidelity, factual consistency, and strict length constraints ($\pm 20\%$ of the original word count) (for prompts see Appendix C).

Best-of-N Random Search (BNRS) Our second baseline introduces a derivative-free search mechanism. This baseline utilizes the same context-conditioned prompt structure as the zero-shot baseline, but explicitly introduces randomized stylistic guidance. For a given text and its associated metadata context, we randomly sample a composite strategy from the predefined library of categorical traits $\mathcal{S}$. We generate $N = 20$ distinct stylistic variations per text using a high LLM temperature setting to encourage diversity (we limit the base-

line to $N = 20$ because larger random allocations yield plateauing rewards). The variants are then scored by the objective function and the variant that achieves the highest predicted reward score is selected as the final output (for prompts see Appendix C).

4.2.2 Metrics

To evaluate the performance of OOCO, we employ two sets of metrics that balances objective maximization with the preservation of document integrity (see Table 1). The first set, termed Reward Optimization Metrics, quantifies the framework's effectiveness in enhancing the specific behavioral objective defined by the objective function. These are measured through the Mean Improvement Percentage, which captures the average gain in predicted reward across the corpus, and the Percentage of Improved Texts, which tracks the reliability of the system in producing superior candidates compared to the source. The second set, Constraint Satisfaction Metrics (see Appendix F), serves as a multi-dimensional audit of the rewritten text's quality. This suite includes the Bidirectional Entailment Score (BES) (Ahn and Khosmood, 2022; Park and Cho, 2025) to enforce semantic fidelity and prevent hallucinations, the Length Ratio (Wu and Deng, 2025; Colombo et al., 2022) to ensure structural adherence, and measures of Perplexity (Kann et al., 2018) and Coherence (Graesser et al., 2004) to guarantee that the final output remains linguistically natural and logically consistent.

4.3 Results

The results of the experiments are reported in Table 2. We ran GSM with both gradient mapping mechanisms (Cosine and Interpreter), but ran GGES only with the Cosine mechanism (given that it already performs extensive manifold exploration, pairing the framework with a closed-form geometric alignment is sufficient to evaluate its performance without introducing parametric complexity). We used Llama-3.3-70B-Instruct (Dubey et al., 2024) and GPT-OSS-120B (OpenAI, 2025) as our rewriting LLM agents. To see a rewriting example, refer to Appendix G.

Main Optimization Results Across both evaluation domains, all three OOCO approaches—GSM-Cosine, GSM-Interpreter, and GGES—outperform the baselines. On the Mashable virality dataset, the proposed methods achieve mean reward improvements ranging from 7.15% to 22.10%, compared to 2.98%–5.20% for the baselines. The lowest-performing OOCO configuration (GSM-Interpreter with Llama, 7.15%) exceeds the strongest baseline (BNRS with GPT, 5.20%) by a factor of $1.4\times$. The peak variant on this dataset is GGES with GPT, yielding a 22.10% reward improvement with 91.3% text coverage.

On the marketing email dataset, the ZSCCP baselines max out at 18.55%, whereas the BNRS baseline achieves a competitive 59.07%–60.47% mean improvement. This baseline outperforms all Llama-based OOCO variants (25.12%–54.92%) and GGES with GPT (49.44%–59.49%). However, when paired with GPT, the iterative approaches outpace BNRS; GSM-Cosine reaches 81.93% and GSM-Interpreter reaches 80.99%, an absolute gain of over 20 percentage points above random search. Notably, GSM-Interpreter consistently outperforms GSM-Cosine when paired with GPT on Mashable (19.75% vs. 18.48%) and Llama on email (54.92% vs. 49.04%).

To verify that reward improvements reflect genuine content quality rather than exploitation of the primary objective function, we conduct a cross-architecture ablation using an Independent Grader — an XGBoost model trained on the identical feature space. All rewritten outputs are scored on this second evaluator without further optimization; transferability of gains confirms that optimization targets robust linguistic features. On Mashable, OOCO methods retain 2.8%–4.8% improvement on the Independent Grader (led by GSM-Interpreter with GPT at 4.8%), while baselines degrade to 0.2%–0.3%. On the email task, GSM-Interpreter maintains 38.5% mean improvement with close cross-evaluator alignment. GGES exhibits a wider disparity between the two evaluators.

Constraint Satisfaction and Text Quality Fidelity indicators reveal distinct operational boundaries across configurations. GGES with Llama achieves the highest Bidirectional Entailment Score (BES) across both datasets (0.845 on Mashable; 0.950–0.951 on email). In contrast, GPT configurations report lower BES ranges (0.594–0.654 on Mashable; 0.874–0.885 on email). Perplexity tracking remains highly stable across all experimental runs, bounding tightly between 0.430 and 0.534.

For the email corpus, the coherence scores of the OOCO methods (0.359–0.402) systematically exceed the BNRS baseline (0.340–0.354) but remain below the ZSCCP Llama baseline (0.437). Length

ratio tracking highlights distinct model characteristics on Mashable, where GPT produces ratios of 0.558–0.594 and Llama yields ratios above 0.914. Within the email environment, length variations narrow significantly for both architectures, tracking at 0.828–0.852 for Llama and 0.848–0.923 for GPT.

5 Conclusions and Discussion

The experimental results provide strong but highly nuanced support for OOCO, showing how reward surface topology and model capacity interact during constrained text optimization. On the Mashable dataset, the evidence is unequivocal: every OOCO configuration outperforms every baseline, establishing the general success of both the OOCO formulation, and the solution approaches.

However, the marketing email dataset presents a more complex optimization landscape where the BNRS baseline proves remarkably competitive. This discrepancy highlights a fundamental difference in the reward surfaces of the two domains. The email reward function is highly responsive to raw rhetorical modifications and text volume, allowing unconstrained stochastic exploration (BNRS) to find high-reward regions through pure volume. Against this highly competitive random walk, gradient guidance's marginal value is strictly contingent on the capacity of the underlying rewriting model. Low-capacity models lack the nuance to execute fine-grained gradient strategies, causing Llama-based OOCO configurations to fall below random search. Conversely, high-capacity models paired with our iterative methods (GSM-Cosine and GSM-Interpreter with GPT) decisively crack this surface, beating random search by over 20 percentage points and proving that directed optimization becomes mandatory once stochastic baselines hit a complexity plateau.

The constraint satisfaction metrics show that OOCO's core contribution is not merely maximizing a raw objective, but navigating optimization trajectories without inducing reward hacking. The high semantic preservation (BES) and length maintenance achieved by Llama across both datasets confirm that our structural safeguards prevent the framework from defaulting to trivial, degenerative text expansions. When aggressive optimization occurs—as seen in the lower BES scores and distorted length ratios of GPT on Mashable—it reflects a model-specific tendency to introduce aggressive rhetorical transformations. Interestingly, this trade-off is deeply domain-dependent: the narrowing of the BES gap and the stabilization of length ratios on the email corpus indicate that source document complexity and inherent domain rigidity can naturally shield text from structural degradation, insulating the output even during intense optimization loops.

The independent grader validation further unpacks the algorithmic divergence between our two paradigms. Because GSM-Interpreter performs reliably across the two evaluators, it likely targets genuine linguistic improvements rather than overfitting to a single architecture. Conversely, GGES trades this cross-architecture consistency for localized, peak optimization. Because GGES relies on an expansive continuous-space scouting phase, its trajectories occasionally isolate evaluator-specific artifacts, resulting in a wider numerical gap between the primary reward model and the independent grader. Nevertheless, the fact that OOCO approaches superior text coherence over random search across both datasets validates that gradient-guided paths structurally anchor the output, ensuring that directed optimization preserves logical flow better than unconstrained exploration.

Limitations

While the findings of the experiments solidify a robust foundation for OOCO, several limitations open compelling vectors for future research. First, our optimization relies entirely on trained objective functions that approximate human behavioral responses; a vital next step is live A/B deployment to confirm that these predicted reward gains translate into real-world behavioral changes. Second, the robust performance of random search on the email dataset underscores the need to formally map reward surface complexity, defining the precise boundary where stochastic exploration fails and gradient-guided optimization becomes necessary. Third, the persistent tension between objective maximization and semantic preservation highlights the need to enforce constraints within the mathematical formulation itself (e.g., via Lagrangian multipliers) rather than relying solely on prompt-level instructions. Finally, expanding the fixed strategy library via data-driven discovery of novel rhetorical dimensions will allow the framework to generalize across high-stakes domains like educational scaffolding and public health communication.

Ethics and Broader Impacts

While Objective-Oriented Content Optimization (OOCO) is designed to enhance benign communication tasks like generalized engagement or personalized marketing, we acknowledge its potential dual-use risks. If paired with malicious objective functions, this framework could theoretically be exploited to optimize for sensationalism, generate toxic clickbait, or craft highly manipulative political misinformation. To mitigate these risks, any deployment of instance-level content optimization should integrate strict semantic fidelity audits—such as our proposed bidirectional entailment constraints—to prevent text distortion, alongside rigorous data governance to protect user privacy in personalized settings.

References

J. Ahn and F. Khosmood. 2022. Evaluation of automatic text summarization using synthetic facts. *arXiv preprint arXiv:2204.04869*.

Yonatan Belinkov. 2022. Probing classifiers: Promises, shortcomings, and advances. *Computational Linguistics*, 48(1):207–219.

Robert B Cialdini. 2001. *Influence: Science and Practice*. Allyn and Bacon.

Pierre Colombo, Chlo'e Clavel, and Pablo Piantanida. 2022. Infolm: A new metric to evaluate summarization data2text generation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pages 10554–10562.

Abhimanyu Dubey, Aaron Grattafiori, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, Amy Yang, Angela Fan, Anirudh Goyal, Anthony Hartshorn, Aobo Yang, Archi Mitra, Archie Sravankumar, Artem Korenev, Arthur Hinsvark, and 7 others. 2024. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*.

Esin Durmus and Claire Cardie. 2018. Exploring the role of prior beliefs for argument persuasion. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics (NAACL)*.

Kelwin Fernandes, Pedro Vinagre, and Paulo Cortez. 2015. A proactive intelligent decision support system for predicting the popularity of online news. In *Progress in Artificial Intelligence: 17th Portuguese Conference on Artificial Intelligence, EPIA 2015*, pages 535–546. Springer International Publishing.

Jessica Ficler and Yoav Goldberg. 2017. Controlling linguistic style aspects in neural text generation. *arXiv preprint arXiv:1707.02633*.

Albert Gatt and Emiel Krahmer. 2018. Survey of the state of the art in natural language generation: Core tasks, applications and evaluation. *Journal of Artificial Intelligence Research*, 61:65–170.

Arthur C. Graesser, Danielle S. McNamara, Max M. Louwerse, and Zhiqiang Cai. 2004. Coh-Metrix: Analysis of text on cohesion and language. *Behavior Research Methods, Instruments, Computers*, 36(2):193–202.

Tatsunori B Hashimoto, Hugh Zhang, and Percy Liang. 2019. Unifying human and statistical evaluation for natural language generation. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics (NAACL)*.

Katharina Kann, Sascha Rothe, and Katja Filippova. 2018. Sentence-level fluency evaluation: References help, but can be spared! In *Proceedings of the 22nd Conference on Computational Natural Language Learning (CoNLL)*, pages 313–323.

Chin-Yew Lin. 2004. Rouge: A package for automatic evaluation of summaries. In *Text Summarization Branches Out*.

Jiahong Liu and 1 others. 2025. Steerx: Disentangled steering for llm personalization. *arXiv preprint arXiv:2510.22256*.

Jekaterina Novikova, Ondřej Dušek, Amanda Cercas Curry, and Verena Rieser. 2017. Why we need new evaluation metrics for nlg. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

OpenAI. 2025. GPT-OSS-120B & GPT-OSS-20B model card. *arXiv preprint arXiv:2508.10925*.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, and 1 others. 2022. Training language models to follow instructions with human feedback. In *Advances in Neural Information Processing Systems (NeurIPS)*.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics (ACL)*.

J. W. Park and K. Cho. 2025. Efficient semantic uncertainty quantification in language models via diversity-steered sampling. In *Proceedings of the 39th Conference on Neural Information Processing Systems (NeurIPS)*.

Rafael Rafailov, Archit Sharma, Eric Mitchell, Stefano Ermon, Christopher D Manning, and Chelsea Finn. 2024. Direct preference optimization: Your language model is secretly a reward model. In *Advances in Neural Information Processing Systems (NeurIPS)*.

Nils Reimers and Iryna Gurevych. 2019. Sentence-BERT: Sentence embeddings using siamese BERT-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3982–3992. Association for Computational Linguistics.

Ehud Reiter and Anja Belz. 2009. An investigation into the validity of some metrics for automatically evaluating natural language generation systems. *Computational Linguistics*, 35(4):529–558.

Alireza Salemi, Sheshera Mysore, Michael Bendersky, and Hamed Zamani. 2024. Lamp: When large language models meet personalization. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (ACL)*.

Kaitao Song, Xu Tan, Tao Qin, Jianfeng Lu, and Tie-Yan Liu. 2020. MPNet: Masked and permuted pre-training for language understanding. *Advances in Neural Information Processing Systems (NeurIPS)*, 33:16857–16867.

Likang Wu, Zhihui Zheng, Zhaopeng Qiu, Hongshuo Wang, Honglian Wei, Jia Guo, Wenqiang Zhu, Quanyu Miao, Houqiang Zhu, Chen Chen, and 1 others. 2023. A survey on large language models for recommendation. *arXiv preprint arXiv:2305.19860*.

Y. Wu and X. Deng. 2025. Implementing long text style transfer with LLMs through dual-layered sentence and paragraph structure extraction and mapping. *arXiv preprint arXiv:2505.07888*.

Chuhan Zhang and 1 others. 2024. Democratizing large language models via personalized parameter-efficient fine-tuning. *arXiv preprint arXiv:2402.04401*.

Yongchao Zhou, Andrei Ioan Muresanu, Ziwen Han, Keiran Paster, Silviu Pitis, Harris Chan, and Jimmy Ba. 2023. Large language models are human-level prompt engineers. In *The Eleventh International Conference on Learning Representations (ICLR)*.

A The Rewriting Strategy Library S

The full taxonomy of the rewriting strategy library is provided in Table 3

B Interpreter Model: Architecture and Training Details

One of the two mechanisms for translating gradients into rewriting instructions in this paper is an "Interpreter Model". To train this model to accurately identify the optimal rewriting instructions, we required a large-scale, highly diverse parallel corpus of original and stylistically altered texts. This section details the data collection, synthetic generation, and architectural hyperparameters for reproducibility.

B.1 Data Collection and Length-Stratified Sampling

To ensure domain-agnostic generalization, base texts were aggregated from diverse linguistic sources, deliberately chosen to expose the model to distinct gradient topographies. This corpus includes short, informal conversational text (`everyday_language`[2]), specialized technical topics and news (`20_newsgroups`[3]), formal academic and encyclopedic entries (`wikitext-2`[4]), and descriptive narratives tailored for shifting authorial voice (`pastel_train` and `pastel_paragraph`[5]).

After standardizing the text, we implemented a rigorous length-stratified sampling methodology. The corpus was divided into six predefined word-count buckets: 15–50, 51–100, 101–200, 201–300, 301–400, and >400 words. We sampled exactly 2,000 records from each of the larger buckets, alongside 1,000 records from the 15–50 word bucket. This yielded a balanced baseline dataset of 11,000 texts (T_{base}), guaranteeing that the Interpreter Model learns to decode pure semantic vector shifts rather than relying on length variations as a confounding proxy for style.

B.2 Learning Strategic Instructions via Synthetic Data

To train the Interpreter Model to map high-dimensional gradients to the categorical stylistic traits within our strategy library $\mathcal{S}$, we leveraged an LLM to construct a parallel corpus of original texts (T_{base}) and strategically altered synthetic texts (T_{syn}).

The generation process strictly enforced semantic fidelity, explicitly instructing the LLM to preserve core factual meaning while exclusively altering the rhetorical presentation (via the prompt template in Appendix C). To ensure the Interpreter Model could untangle complex, overlapping signals, the dataset was constructed in two phases:

Phase A (Isolated Main Effects): To establish baseline vector representations, we generated 100,000 synthetic rewrites by applying exactly one trait from the pool of 50 available adjectives in $\mathcal{S}$. By producing 2,000 texts per trait, this phase

[2] https://huggingface.co/datasets/MultivexAI/Everyday-Language-Corpus

[3] https://huggingface.co/datasets/SetFit/20_newsgroups

[4] https://huggingface.co/datasets/Salesforce/wikitext

[5] https://github.com/dykang/PASTEL

Category	Adjective	Operational Definition / Style Guide
The "Vibe" & Tone	Bold	Use strong, confident statements; strictly avoid hedging.
	Playful	Adopt a lighthearted, witty tone with engaging, conversational phrasing.
	Subtle	Deliver messages with nuance, restraint, and understated styling.
	Clinical	Use objective, detached, and factual language free of emotional bias.
	Formal	Use elevated, professional, and grammatically strict language.
	Authoritative	Write with commanding expertise to sound like a definitive source of truth.
	Visceral	Use vivid, emotionally charged, and sensory-rich language.
	Intellectual	Elevate prose with academic phrasing and sophisticated vocabulary.
	Warm	Use an inviting, empathetic, and friendly conversational tone.
	Dry	Deliver matter-of-fact, understated statements with a touch of deadpan.
Clarity & Precision	Lucid	Ensure text is highly transparent, clear, and instantly understandable.
	Precise	Use exact terminology and concrete details to eliminate all ambiguity.
	Granular	Focus heavily on minute details and highly specific elements.
	Holistic	Highlight the big-picture view, connecting details to overarching themes.
	Concise	Deliver ideas briefly and directly; eliminate unnecessary filler words.
	Direct	Address the main points head-on without circumlocution.
	Oblique	Approach topics indirectly through implication and subtle suggestion.
	Accessible	Frame concepts simply so a layperson can easily comprehend.
	Esoteric	Use highly specialized, domain-specific terminology for niche audiences.
	Cryptic	Use enigmatic, open-ended statements that leave room for interpretation.
Structure & Flow	Dynamic	Vary sentence length and structure to create an active, energetic pace.
	Methodical	Build arguments systematically through a logical, step-by-step structure.
	Linear	Maintain a strict chronological or logical progression without digressions.
	Erratic	Use rapid, unexpected transitions between ideas to create a disjointed flow.
	Cohesive	Ensure sentences are tightly integrated and transition seamlessly.
	Staccato	Employ short, punchy, and abrupt sentences for dramatic emphasis.
	Balanced	Maintain a steady rhythm with symmetric clauses and balanced arguments.
	Organic	Allow ideas to flow naturally, resembling an intuitive train of thought.
	Static	Maintain a highly uniform, predictable sentence structure and pace.
	Circular	Begin and end on the same concept, looping back to the main premise.
Complexity & Depth	Dense	Maximize information density by packing meaning into minimal space.
	Minimalist	Use the absolute bare minimum of words to convey core concepts.
	Nuanced	Capture subtle distinctions, complex gray areas, and multi-faceted views.
	Simple	Keep explanations straightforward and use common, everyday vocabulary.
	Lean	Strip away auxiliary adverbs and adjectives for muscular, efficient prose.
	Rich	Use elaborate, highly descriptive, and evocative phrasing.
	Abstract	Focus on theoretical, conceptual ideas rather than concrete realities.
	Concrete	Anchor arguments in tangible, physical examples and real-world realities.
	Layered	Embed multiple subtexts and interpretative layers within the text.
	Shallow	Maintain focus strictly on the surface level; avoid deep implications.
Innovation & Execution	Radical	Present arguments in a groundbreaking, paradigm-shifting manner.
	Traditional	Adhere strictly to classical styling, established standards, and norms.
	Polished	Produce highly refined, smooth, and meticulously edited prose.
	Raw	Maintain an unrefined, authentic, and emotionally unfiltered language.
	Spontaneous	Give the text an energetic, improvised, and highly natural feel.
	Systematic	Organize concepts into rigid, highly structured, and repeatable frameworks.
	Fresh	Use novel phrasing and modern metaphors; strictly avoid clichés.
	Elegant	Construct graceful, sophisticated sentences that flow effortlessly.
	Functional	Prioritize practical utility and functional clarity over artistic style.
	Disciplined	Exercise strict control over the scope; avoid any stylistic excesses.

Table 3: The full taxonomy of the rewriting strategy library. Concrete actions are generated dynamically by sampling 1 to 5 adjectives, allowing a maximum of one adjective per category.

isolated the unconfounded vector shift of singular instructions (e.g., “Urgent” or “Concise”).

Phase B (Complex Interactions): To account for nuanced relationships across rhetorical categories, we generated an additional 425,000 synthetic rewrites with composite strategies dynamically sampled from $\mathcal{S}$. Each category had an equal probability of either contributing one of its adjectives or remaining inactive, ensuring every applied strategy contained a minimum of two active adjectives (e.g., “Bold” + “Lucid” + “Dynamic”).

The resulting 525,000-record parallel dataset was translated into a trainable numerical format using a pre-trained sentence encoder (`all-mpnet-base-v2`). By calculating the base embeddings ($\mathbf{E}_{\text{base}}$) and isolating the directional change via the delta embedding ($\Delta\mathbf{E} = \mathbf{E}_{\text{syn}} - \mathbf{E}_{\text{base}}$), we provided the MLP with a rich mapping dataset ($D_{\text{interpreter}}$).

Although the dataset used to train the Interpreter was synthetically generated by an LLM, this does not create a closed-loop dependency that limits the framework. The Interpreter is trained strictly to classify the geometric relationship between continuous reward gradients and discrete strategy labels. Because these editing strategies represent universal linguistic concepts, the Interpreter learns general directional mappings rather than the idiosyncratic behaviors of a specific model.

B.3 Interpreter Architecture and Context-Aware Translation

The Interpreter Model employs a Base-Conditioned Concatenation MLP. To evaluate the proposed stylistic shift ($\Delta\mathbf{E}$) within the context of the original text ($\mathbf{E}_{\text{base}}$), the two 768-dimensional embeddings are concatenated into a single 1536-dimensional composite vector. This omits complex attention mechanisms to enforce intrinsic conditional learning.

Formally, let $\oplus$ denote vector concatenation. The forward pass is defined as:

$$\begin{aligned}
\mathbf{x} &= [\mathbf{E}_{\text{base}} \oplus \Delta\mathbf{E}] \in \mathbb{R}^{1536} \\
\mathbf{h}_1 &= \text{Dropout}(\text{ReLU}(\text{BN}(\mathbf{W}_1\mathbf{x} + \mathbf{b}_1))) \\
\mathbf{h}_2 &= \text{Dropout}(\text{ReLU}(\text{BN}(\mathbf{W}_2\mathbf{h}_1 + \mathbf{b}_2))) \\
\hat{\mathbf{y}} &= \mathbf{W}_3\mathbf{h}_2 + \mathbf{b}_3
\end{aligned} \tag{3}$$

where $\mathbf{W}_1 \in \mathbb{R}^{512\times1536}$, $\mathbf{W}_2 \in \mathbb{R}^{256\times512}$, and $\mathbf{W}_3 \in \mathbb{R}^{50\times256}$ are the learned weight matrices. BN denotes 1D Batch Normalization, and the dropout probability is set to $p = 0.2$ to mitigate overfitting. The final logits $\hat{\mathbf{y}} \in \mathbb{R}^{50}$ correspond to the predicted probabilities of the 50 categorical traits within $\mathcal{S}$.

The 525,000-record dataset ($D_{\text{interpreter}}$) was partitioned into standard training (80%), validation (10%), and test (10%) splits. After optimization via the validation set, final predictive capabilities were measured strictly on the unseen test split. Empirical results (Figure 2) demonstrate robust predictive capabilities, though performance variance is contingent upon the semantic distinctiveness of the specific stylistic trait within $\mathcal{S}$.

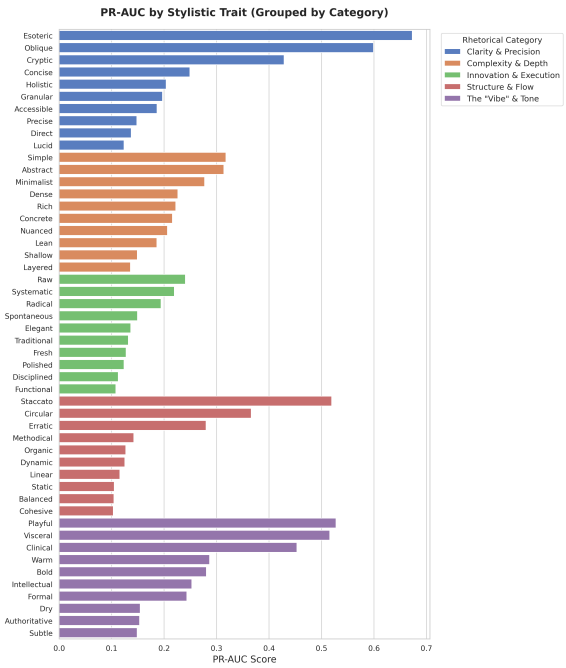

Figure 2: Test-set evaluation of the Interpreter Model across the 50 stylistic traits in $\mathcal{S}$. The model processes the base-conditioned vector shift ($[\mathbf{E}_{\text{base}} \oplus \Delta\mathbf{E}]$) to accurately reverse-engineer rhetorical intent. While overall predictive capability is robust, performance variance across categories highlights that highly distinct rhetorical styles are easier for the model to isolate than highly nuanced or overlapping traits.

C Prompt Templates

Prompt Template for Generating 'KEY POINTS FROM ORIGINAL ARTICLE' (the outcome of this LLM call is injected in every rewriting LLM prompt to ensure adherence to the original contents)

Summarize the following article as a bullet-point list of key facts, claims, and themes. Capture every important detail — this summary will be used to verify that future rewrites preserve the original meaning.

ARTICLE:
{article_text}

Provide ONLY the bullet-point summary, no introductory or concluding remarks.

Prompt Template for Iterative Content Rewriting

The following bullet points summarize the key facts, claims, and themes of the original article. Your rewrite MUST preserve ALL of these points — do not drop, contradict, or fabricate any of them.

KEY POINTS FROM ORIGINAL ARTICLE:
{bullet_summary}

ORIGINAL WORD COUNT: {original_word_count}

—

Rewrite the following text to make it more engaging based on the STYLE GUIDANCE while strictly adhering to the original meaning.

TEXT TO REWRITE:
{current_text}

STYLE GUIDANCE:
{% for cat, adj, desc in selected_items -%}
- {cat} ({adj}): {desc}
{% endfor %}

RULES:
- **Maintain Fidelity:** Do not invent new details that the original author wouldn't recognize it.
- **Surgical Improvements:** Only change words or sentence structures that directly benefit from the Style Guidance. If a sentence is already effective, leave it alone.
- **Constraint over Creativity:** The core facts and topic are non-negotiable. Do not add details that weren't in the source.
- **Length:** Stay within $\pm 20\%$ of the original word count ({original_word_count} words). Your rewrite should be between {min_words} and {max_words} words. Keep the same number of paragraphs plus/minus 1 as the original text.
- **Natural Clarity:** The text must be easy for a human to read on the first pass. Avoid overcomplicated sentence structures or dense phrasing. If a rewrite sounds like "word salad" or gibberish, discard it and try a simpler approach.
- **Output:** Provide ONLY the rewritten text without any introductory or concluding remarks.

IMPORTANT CORRECTION: A previous rewrite of this text was rejected because it drifted from the original meaning. The specific issue was:
{fidelity_feedback}

You MUST avoid this problem. Apply the stylistic changes above while strictly preserving the original facts, claims, and meaning. Do not introduce any new information or change the subject matter.

Prompt Template for Semantic Fidelity Check

You are a fact-checking editor. Compare the ORIGINAL and REWRITTEN texts below.

ORIGINAL:
{original_text}

REWRITTEN:
{rewritten_text}

Does the rewritten text preserve the same core facts, claims, and overall meaning as the original? Minor stylistic changes are acceptable, but the rewrite must NOT:
- Add new facts or claims not in the original
- Remove or contradict any key information
- Change the topic or subject matter
- Significantly alter the meaning of any statement
- Have overcomplicated sentence structures or dense phrasing
- Sound like "word salad" or gibberish

Answer with ONLY "PASS" or "FAIL" on the first line, followed by a one-sentence explanation.

Prompt Template for Unified Baseline Benchmark

[Objective for Mashable Virality Environment:]
Rewrite the following text to maximize its virality and broad audience appeal, while strictly adhering to the original meaning.

[Objective for Email Engagement Environment:]
Rewrite the following text to make it more engaging and personalized to the provided recipient profile, while strictly adhering to the original meaning.

[Injected for Email Engagement Environment Only]
TARGET PROFILE:
To help you tailor the tone, urgency, and complexity, here is the profile compared to the average population (Mean $\pm$ Standard Deviation):
{context_str}

[Injected for Random Search Baseline Only]
STYLE GUIDANCE:
You must flawlessly adopt the following stylistic attributes in your rewrite:
{style_instructions}

TEXT TO REWRITE:
{current_text}

RULES:
- **Maintain Fidelity:** Do not invent new details that the original author wouldn't recognize.
- **Surgical Improvements:** Only change words or sentence structures that directly adapt the text to the optimization goal [and Style Guidance]. If a sentence is already effective, leave it alone.
- **Constraint over Creativity:** The core facts and topic are non-negotiable. Do not add details, product offers, or features that weren't in the source.
- **Length:** Stay within $\pm 20\%$ of the original word count ({original_word_count} words). Your rewrite should be between {min_words} and {max_words} words. Keep the same number of paragraphs plus/minus 1 as the original text.
- **Natural Clarity:** The text must be easy for a human to read on the first pass. Avoid overcomplicated sentence structures or dense phrasing.
- **Output:** Provide ONLY the rewritten text without any introductory or concluding remarks.

Prompt Template for Synthetic Generation

Rewrite the following source by adopting the specific stylistic traits listed below. Make only targeted improvements – do not change the topic, facts, or overall structure:
{style_guidelines}

Guidelines to follow:
1. Ensure the core content, facts, and original meaning of the source remain completely unaltered.
2. The length of the new text should be roughly the same as the original.
3. The structure of individual parts (such as line spacing, bullet points) must remain intact.
4. Do not omit, add, or hallucinate any core information. Your goal is to purely change the stylistic presentation according to the traits above.
5. Output ONLY the rewritten text.
Source: {text}

D Mashable News Virality Data and Model

D.1 Dataset

Our first experiment is based on the Mashable Online New Popularity corpus (Fernandes et al., 2015), obtained from the UCI Machine Learning Repository. This dataset provides a large-scale record of real world content performance of about 39,000 articles published on Mashable.com between 2013 and 2015, each annotated with a social media share count as a proxy for audience engagement. We randomly sample 23,000 articles via webscraping and after filtering articles without text, with zero shares, or with zero keyword metrics, we split 18,347 into a train and test set by proportion 70:30.

D.2 Labeling Strategy and Objective Function

An objective function is needed to optimize text based on and to evaluate each rewrite given text and metadata relating to it. We deliberately normalize share counts — as these counts exhibit severe right-skew and high variance given viral outliers — by creating a binary label with 0 being not viral and 1 being viral. Raw share counts (s_i) are normalized according to:

$$\hat{s}_i = log(1 + \sqrt{\frac{s_i}{kw_avg_i \times reference_shares_i}}) \tag{4}$$

This formulation serves two purposes. The log compression reduces sensitivity to extreme values in the tail of the distribution. The denominator removes systemic variance attributable to topic popularity (kw_avg_i) and author-level engagement history ($reference_shares_i$) rather than intrinsic content quality. The geometric mean is the appropriate normalizer under multiplicative assumption: both baseline factors are assumed to scale share counts proportionally, so their joint contribution is captured by their product, and normalization requires the square root of that product. We classify any share count $\hat{s}_i \geq 0.5$ is viral, otherwise it is not viral.

D.3 Feature Engineering

The goal of the objective function is to predict the likelihood that a given piece of content will be viral. The objective function receives two inputs – a list of features (15D) and a piece of content (768D) – concatenated into a 783-dimensional vector.

Text Representation: Article body text is encoded using using a pre-trained Sentence-Transformer, `all-mpnet-base-v2` (Reimers and Gurevych, 2019; Song et al., 2020), that produces dense, semantically rich 768-dimensional representations. This embedding serves as the primary variable manipulated by our rewriting loop.

Metadata Features: The top-15 most predictive UCI features are selected via prior feature important analysis and appended to the text embedding as a fixed context vector. These features can be seen in Table 4. We chose 15 features empirically

The separation of text embedding from metadata is architecturally significant: metadata features are fixed per article across rewriting iterations, while the text embedding is the sole variable input under optimization. This design ensures that the reward gradient is computed exclusively with respect to the semantic content of the text, not confounded by static structural properties.

D.4 Objective Function Architecture and Training

We train a shallow multi-layer perceptron (MLP) as the differentiable objective function $R(x, z)$ where $x \in \mathbb{R}^{768}$ is the text embedding and $z \in \mathbb{R}^{15}$ is the fixed metadata vector:

$$\begin{aligned} \mathbf{v} &= [\mathbf{x} \parallel \mathbf{z}] \in \mathbb{R}^{783} \\ \mathbf{h}_1 &= \text{Dropout}_{0.3}(\text{ReLU}(\text{BN}(\mathbf{W}_1\mathbf{v} + \mathbf{b}_1))) \\ \mathbf{h}_2 &= \text{Dropout}_{0.3}(\text{ReLU}(\text{BN}(\mathbf{W}_2\mathbf{h}_1 + \mathbf{b}_2))) \\ \hat{p} &= \sigma(\mathbf{W}_3\mathbf{h}_2 + \mathbf{b}_3) \end{aligned} \tag{5}$$

where $\mathbf{W}_1 \in \mathbb{R}^{256\times796}$, $\mathbf{W}_2 \in \mathbb{R}^{64\times256}$, and $\mathbf{W}_3 \in \mathbb{R}^{1\times64}$ denote the learnable weight matrices, and $\hat{p} \in (0, 1)$ represents the model's estimated probability that the article falls above the 50th percentile of normalized share count. The shallow architecture (two hidden layers) is intentional: models with limited depth produce smoother, better-conditioned gradients with respect to their input, which is a prerequisite for reliable gradient-guided optimization. Deeper networks with additional ReLU nonlinearities introduce more discontinuities in the input-gradient landscape, making the per-step optimization direction less stable.

Training uses Binary Cross-Entropy loss with Adam optimization ($lr = 10^{-3}$, weight decay $= 10^{-3}$), batch size 256, and early stopping with patience of 7 epochs on validation loss (maximum 50 epochs). The model achieves a test-set ROC-AUC of 0.784, confirming sufficient discriminative power to provide a meaningful optimization signal.

D.5 Feature Selection

The UCI Online News Popularity dataset provides 58 pre-computed features spanning keyword performance metrics, self-referential sharing patterns, topic distributions (LDA), sentiment analysis, text complexity measures, content channel indicators, and publication timing. To construct the reward model's metadata input, we select 15 of these 58 features through a two-stage filtering process combining empirical importance ranking with an actionability criterion specific to our optimization setting.

Feature	Description	Rationale
kw_avg_avg	Mean shares: articles with same best keywords.	Proxy for topic-level distribution potential.
self_ref_avg	Mean shares of referenced articles.	Controls for outlet/author baseline audience.
kw_avg_max	Mean of max shares among keyword groups.	Upper bound of topic reach.
kw_max_avg	Max keyword group's avg. shares.	Strongest keyword signal.
global_subj	Text subjectivity score [0,1].	Subjective content provokes sharing.
self_ref_min	Min. shares of referenced articles.	Floor of outlet audience.
LDA_01	Closeness to LDA topic 1.	Latent topic distribution.
non_stop_uniq	Rate of unique non-stop words.	Lexical diversity.
kw_avg_min	Mean of min. shares among keyword groups.	Lower bound of reach.
avg_token_len	Mean word length (chars).	Correlates with reading difficulty.
LDA_02	Closeness to LDA topic 2.	Latent topic distribution.
LDA_04	Closeness to LDA topic 4.	Latent topic distribution.
n_uniq_tokens	Rate of unique words / total words.	Vocabulary richness.
LDA_03	Closeness to LDA topic 3.	Latent topic distribution.
n_tokens_cont	Total words in article.	Content length signal.

Table 4: Key Features and Rationale

We first rank all 58 features by importance from a Random Forest classifier trained on the full feature set (ROC-AUC = 0.807 with all features). As shown in Figure 3, self-reference metrics, keyword averages, text complexity measures, and LDA topic closeness scores consistently occupy the top ranks of this importance ordering, with our 15 selected features (highlighted in blue) concentrated in the

upper half of the ranking. We then apply a second filter: because the reward model serves as a differentiable objective for text optimization—not merely a standalone classifier—we retain only features whose underlying signal is plausibly influenced by text re0writing. Text complexity features (word count, vocabulary richness, token length) change directly with every rewrite. LDA topic closeness shifts as the LLM adjusts emphasis and framing. Global subjectivity responds to tone and style modifications. Keyword and self-reference features, while fixed per article, provide essential calibration context that allows the model to distinguish "high quality for a difficult topic" from "mediocre quality on an easy topic." We exclude all channel indicators (6 features), publication timing (8 features), structural content metrics (image/video/link counts, 5 features), and granular sentiment breakdowns (15 features)—these are either immutable properties of the article's distribution context or provide redundant signal already captured by the single global subjectivity feature.

A feature count ablation (Figure 4) validates this selection empirically. Performance increases sharply from 3 to 8 features (AUC: 0.735 → 0.780), flattens through the 8–15 range (0.780 → 0.784), and continues a slow, noisy climb through 58 features (0.784 → 0.807). The marginal gain per feature drops from approximately 0.015 AUC/feature in the 3–8 range to 0.0005 AUC/feature in the 15–58 range—a 30× reduction in efficiency. The additional 2.3 AUC points available from features 16–58 come entirely from non-actionable dimensions (channel, timing, media counts) that would inject gradient signal in directions the text optimizer cannot actualize, potentially degrading optimization efficiency while increasing overfitting risk on the 12,842-sample training set. The final 783-dimensional input (768D text embedding + 15D metadata) achieves an ROC-AUC of 0.784 on the held-out test set, providing sufficient discriminative power for gradient-based content optimization while maintaining a clean, actionable reward signal.

E Marketing Email Data and Model

E.1 Dataset

Our second experiment is focused on marketing email hyper-personalization. We used a proprietary dataset of historical user interactions (click-throughs) with marketing emails. To isolate true

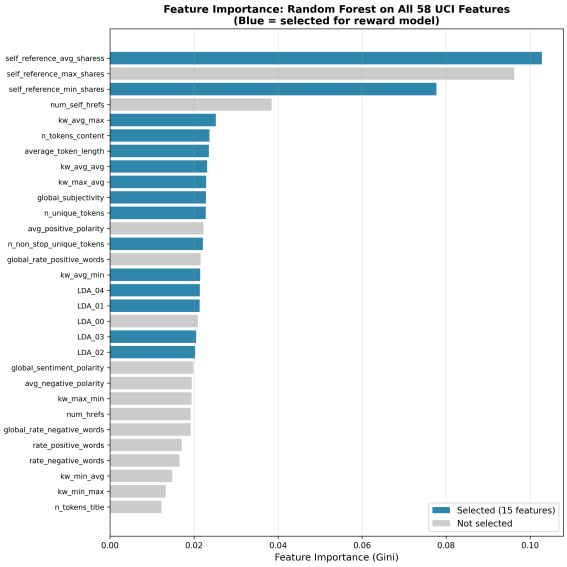

Figure 3: Random Forest feature importance (Gini) across all 58 UCI features. The 15 features selected for the reward model (blue) are concentrated in the top ranks. Excluded features (gray) are predominantly non-actionable properties such as content channel, publication timing, and media counts that cannot be influenced through text rewriting.

engagement driven by content rather than external factors (such as subject line appeal), the dataset was constrained to emails that were successfully opened by the recipient.

The primary optimization objective—and the target label for our objective function—is a binary classification task: whether the user clicked a link within the email (*clicked* = 1) or did not (*clicked* = 0). Because email click-through rates are inherently skewed toward the negative class, addressing severe class imbalance was a central consideration in both the data preparation and objective function design.

E.2 Feature Engineering

Similar to the first experiment, to capture the complex interplay between the email content and the recipient's profile, the model ingests two distinct feature sets: unstructured text and structured tabular metadata.

Text Representation: The raw body of the email was mapped into a continuous vector space using a pre-trained SentenceTransformer `all-mpnet-base-v2` (Reimers and Gurevych, 2019; Song et al., 2020). This generated dense, 768-dimensional semantic embeddings capable of capturing the stylistic and thematic nuances of the

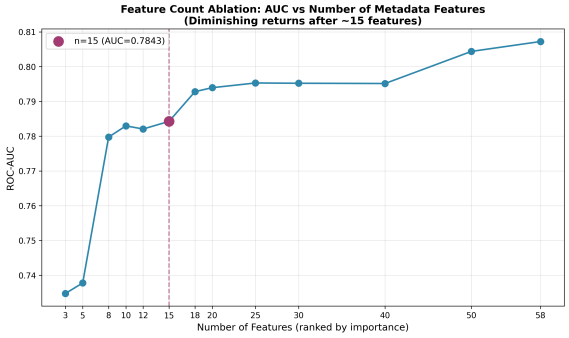

Figure 4: Feature count ablation showing ROC-AUC as a function of the number of metadata features included (ranked by importance). Performance rises sharply from 3 to 8 features, plateaus through 15, and gains only 2.3 additional AUC points over the remaining 43 features. The selected cutoff at $n = 15$ (marked) captures the transition from high-marginal-gain features to diminishing returns dominated by non-actionable dimensions.

text.

Tabular Data Transformation: The proprietary numerical metadata required specialized imputation and scaling to handle missingness and varied statistical distributions. We utilized a bifurcated preprocessing pipeline via a `ColumnTransformer`. All tabular features were standardized using a standard scaler (zero mean, unit variance) to ensure stable gradient flow during network training.

The proprietary marketing dataset contains hundreds of candidate features spanning demographic profiles, real-time transaction histories, and historical communication engagement. To construct the reward model's tabular input (D_{tab}), we select 28 specific features through a filtering process optimized for our early-concatenation neural architecture. Due to confidentiality requirements, more details on the features and feature selection cannot be provided here. The 28 selected features provide the neural network with the exact behavioral priors necessary to evaluate how a specific user profile will react to the 768-dimensional text embedding.

E.3 Objective Function Architecture and Training

Similar to the first experiment, we train a shallow MLP as the differentiable objective function $R(x, z)$, where $x \in \mathbb{R}^{768}$ is the email text embedding and $z \in \mathbb{R}^{28}$ is the preprocessed, fixed tabular user persona vector. The modalities are fused via early concatenation:

$$\begin{aligned} \mathbf{v} &= [\mathbf{x} \parallel \mathbf{z}] \in \mathbb{R}^{796} \\ \mathbf{h}_1 &= \text{Dropout}_{0.3}(\text{ReLU}(\text{BN}(\mathbf{W}_1\mathbf{v} + \mathbf{b}_1))) \\ \mathbf{h}_2 &= \text{Dropout}_{0.3}(\text{ReLU}(\text{BN}(\mathbf{W}_2\mathbf{h}_1 + \mathbf{b}_2))) \\ \hat{p} &= \sigma(\mathbf{W}_3\mathbf{h}_2 + \mathbf{b}_3) \end{aligned} \tag{6}$$

where $\mathbf{W}_1 \in \mathbb{R}^{256\times796}$, $\mathbf{W}_2 \in \mathbb{R}^{64\times256}$, and $\mathbf{W}_3 \in \mathbb{R}^{1\times64}$ denote the learnable weight matrices, and $\hat{p} \in (0, 1)$ represents the predicted probability that the user engages with a link within the opened email. And to address the heavily imbalanced click data, the network is trained using Binary Cross-Entropy with Logits Loss ($\mathcal{L}_{BCE}$), scaled by a dynamic positive weight multiplier ($w_p = \frac{N_{\text{negative}}}{N_{\text{positive}}}$). Training utilizes the AdamW optimizer ($lr = 10^{-3}$, weight decay $= 10^{-4}$) with a batch size of 256 for 15 epochs. Model checkpointing is tied directly to the validation PR-AUC to preserve the highest sensitivity to the rare positive class.

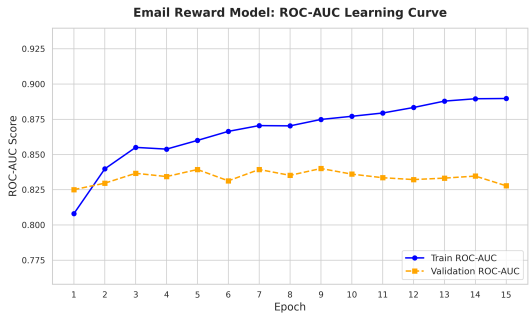

Figure 5: Epoch-by-epoch ROC-AUC learning curve for the Marketing Email Objective Function.

As illustrated in Figure 5, the model demonstrated strong and rapid convergence. The optimal weights were captured at Epoch 5. When strictly evaluated on the unseen test dataset, this optimal checkpoint achieved a final ROC-AUC of 0.8394 and a PR-AUC of 0.0917, confirming robust ranking capabilities above background noise.

F Constraint Satisfaction Metrics

To properly and rigorously evaluate the rewritten text against the original source, we define a suite of post-processing metrics that collectively measure semantic fidelity, structural adherence, linguistic naturalness, and coherence in the language. Each metric captures a distinct component of quality degradation or improvement, and together they provide a comprehensive evaluation framework that resists reward hacking any single objective. We describe each metric used below including its formal

definition, why it was chosen, its computational procedure, and its role in the evaluation pipeline.

F.1 Bidirectional Entailment Score

A key requirement of faithful text writing is that the output neither omits concepts presented in the source (information loss) nor introduces unsupported claims (hallucination). We can use concepts from Natural Language Inference (NLI) to determine whether one text logically and correctly entails another and see whether content has been preserved, added or removed.

We define the Bidirectional Entailment Score (BES) as the harmonic mean of forward and backward entailment probabilities between the source text X and the rewritten text Y.

$$\begin{aligned} E_{\text{fwd}} &= M_{\text{NLI}}(X, Y) \\ E_{\text{bwd}} &= M_{\text{NLI}}(Y, X) \\ \text{BES}(X, Y) &= \frac{2 \cdot E_{\text{fwd}} \cdot E_{\text{bwd}}}{E_{\text{fwd}} + E_{\text{bwd}}} \end{aligned} \tag{7}$$

A low forward entailment score indicates that the rewritten text Y contains claims not supported by the source text X (hallucination), and a low backward entailment score indicates the original contains information that is not present in the rewrite (omission). The harmonic mean penalizes asymmetric rewritings, ensuring that a rewrite cannot be scored well if only faithful in a single direction.

For documents that exceed the input length of the NLI model, we segment X and Y into sentence-aligned chunks $x_1, ..., x_n$ and $y_1, \ldots, y_n$ and then compute pairwise alignment scores and aggregate using greedy alignment.

$$E_{\text{fwd}} = \frac{1}{n} \sum_{i=1}^{n} \max_{j \in [m]} M_{\text{NLI}}(x_i, y_j) \tag{8}$$

Where E_{bwd} is defined similarly.

The metrics takes as input a pair of texts (X, Y) representing the source text and rewritten text. The output is a scaler, BES, in $[0, 1]$ where 1.0 represents perfect bidirectional entailment (complete semantic equivalence) and values close to 0.0 indicate hallucination, omission, or both.

F.2 Length Ratio

Constrained rewriting tasks frequently impose structural requirements on the output, including approximate length preservation. Without an explicit length constraint, generative models tend towards verbose response, adding redundancy to concise text or excessive compression that often sacrifices detail. A verbose rewrite can get high backward entailment by including everything from the source (introducing filler), and a compressed output might miss key information.

We define the Length Ratio (L_r) as a symmetric, bounded measure of length deviation where:

$$L_r(X, Y) = \min\left(\frac{|Y|}{|X|}, \frac{|X|}{|Y|}\right) \tag{9}$$

Where $| * |$ indicates the length of the text in characters.

This calculation will return a symmetric minimum in the range of $(0, 1]$ where 1 indicates identical length and values closer to 0 indicate proportionally greater deviation. This is one of our less complex metrics, but it serves as a hard constraint filter in our pipeline as each rewrite we ask the model to make should deviate less than 20 percent in terms of total length from the original.

F.3 Perplexity

The two metrics above check more for semantic preservation, but they do not check as much for language that is natural. A rewrite may preserve information from the source, but it may also be awkward, disfluent, or syntactically malformed. This is often prevalent when models optimize aggressively against a semantic objective without a naturalness constraint.

In our case, perplexity, computed using a strong external language model, serves as a way to measure linguistic acceptability - text that is less likely to be seen as unnatural by a human reader.

Given a rewritten text $Y = (y_1, y_2, \ldots, y_T)$ and a reference language model M_{ref}, we define perplexity as:

$$\text{PPL}(Y) = \exp\left(-\frac{1}{T} \sum_{t=1}^{T} \log P_{M_{\text{ref}}}(y_t \mid y_{<t})\right) \tag{10}$$

Essentially, the model will generate the log probabilities of the different words in the sequence based on contextual surroundings, and if the total perplexity is low this indicates the text aligns well with the distributional expectations of the reference model. A locally hosted, frozen version of the GPT-OSS model is accessed to generate log-probability outputs over the target text, and a total value assessing the perplexity of the input as a whole between $[0, 1]$ will be generated, where values closer to 0

indicate less perplexity and closer to 1 indicate more.

To prevent the semantic optimization process from producing grammatically awkward or unnatural rewrites, we utilize sentence-level perplexity via an external reference language model as a proxy for human linguistic acceptability.

F.4 Coherence

Finally, another metric for natural, well-understood text is whether that rewritten text exhibits coherence - or the logical and semantic continuity between adjacent sentences that indicates whether a reader can understand the text without confusion. Rewritten text can often preserve the source information well, yet fail if the ordering, transitions, or referential structure have been disrupted. Particularly in paragraph-level or longer rewriting (tested in our datasets), we can see models rearrange or decontextualize information to hack other rewards.

We measure coherence using sentence-level semantic similarity. Given we generate the rewritten text $Y = (s_1, s_2, \ldots, s_n)$, we first compute the embedding of each sentence using a sentence encoder. From there we measure the cosine similarity between consecutive sentence pairs:

$$\text{Coh}(Y) = \frac{1}{n-1} \sum_{i=1}^{n-1} \cos(\phi(s_i), \phi(s_{i+1})) \quad (11)$$

Where $cos(a, b) = (ab)/(||a|| * ||b||)$ is the cosine similarity function and $\phi(\cdot)$ generates the sentence embeddings.

A high coherence score indicates smooth semantic transitions between sentences, while abrupt changes in local similarity signal topic discontinuities introduced during rewriting. A final score of $[0, 1]$ is returned where values closer to 0 indicate less coherence and values closer to 1 indicate more.

G A Rewriting Example

A rewriting example is provided in Table 5.

H Computing Infrastructure and Budget

The parameter counts for our primary text-rewriting models are 70 billion for Llama-3.3-70B-Instruct and 117 billion for GPT-OSS-120B, while the `all-mpnet-base-v2` text encoder contains 110 million parameters. The custom Multi-Layer Perceptrons (the Interpreter and Surrogate Reward models) are lightweight, scaling under 1 million parameters each. All neural network training pipelines and local inferences were executed on a computing infrastructure utilizing NVIDIA A100-40GB GPUs. The total computational footprint for synthetic parallel corpus generation, interpreter training, and surrogate optimization was approximately 60 total GPU hours.

Iter.	Score	ΔScore	Rewriting Strategies	Rewritten Text
1	0.4505	—	*(original, no rewrite)*	LAS VEGAS — At a private press event Sunday evening, Toshiba gave us a first look at its upcoming line of Ultra HD 4K TVs. The L9300 series is headlined by an 84-inch model (pictured above), but also includes a 58 and 65-inch set. The 4K format gives you four times the pixel resolution of a standard 1080p Full-HD television, offering a picture at 3,840 × 2,160 pixels. Televisions in Toshiba's 4K series use its own CEVO 4K Quad + Dual Core Processor, which converts standard 1080p content to 4K resolution without creating any artifacts in the process. The processor makes it possible to take traditional 1080p content, for instance a Blu-ray disk, and view that content at a near-4K resolution. The processor also boosts the video's contrast. The televisions also take advantage of Toshiba's new Cloud TV platform, giving users access to Skype calling, personal messaging, a family event calendar and streaming news. The L9300 series is expected to ship this summer for a yet-to-be-announced price.
2	0.4630	+0.013	Visceral | Esoteric | Circular | Concrete | Radical	LAS VEGAS — In the hushed glow of a private press gathering Sunday night, Toshiba peeled back the curtain on its next-generation Ultra HD 4K arsenal. The L9300 family storms onto the stage anchored by a monumental 84-inch behemoth, flanked by sleek 58- and 65-inch siblings that promise the same visual ferocity in more intimate formats. The 4K canvas explodes to 3,840 × 2,160 pixels—four-fold the density of conventional 1080p Full-HD—delivering razor-sharp granularity that turns every frame into a tactile tableau. Powering this leap is Toshiba's proprietary CEVO 4K Quad + Dual-Core Processor, a silicon-crafted alchemist that lifts native 1080p sources—Blu-ray discs, streaming files, legacy broadcasts—to near-4K fidelity without spawning the usual ghosting or ringing artifacts. Its contrast-enhancement engine deepens blacks and lifts highlights, sculpting depth that feels almost three-dimensional. Beyond raw pixels, the L9300 line rides on Toshiba's freshly minted Cloud TV platform, a cloud-centric hub that fuses entertainment with connectivity. Users can launch Skype calls from the living-room couch, ping personal messages, sync a family event calendar, and tap streaming news feeds—all without ever leaving the TV's UI. The suite is slated to roll out this summer, with pricing still under wraps, positioning the series as a premium gateway to a cloud-enabled home-theater experience.
			· · · iterations 3–20 omitted · · ·	
21	**0.6185**	**+0.022**	**Bold | Lucid | Dynamic | Dense | Traditional**	**In a hush-filled private press gathering Sunday night, Toshiba ripped the veil from its next-generation Ultra HD 4K arsenal, igniting the electric anticipation that only a true breakthrough can spark. Front-and-center stands an 84-inch titan, flanked by sleek 58-inch and 65-inch siblings that compress identical visual ferocity into tighter footprints. All three belong to the L9300 series, a unified 4K lattice that refuses compromise. The series rests on a concrete pixel matrix of 3,840 × 2,160 pixels—four times the density of conventional 1080p Full-HD—turning every frame into a razor-sharp, tactile tableau that feels as solid as the screen itself. At the heart of this alchemy sits Toshiba's proprietary CEVO 4K Quad + Dual-Core Processor, a silicon-forged engine that ingests native 1080p sources—Blu-ray discs, streaming files, legacy broadcasts—and transmutes them to near-4K fidelity while suppressing ghosting, ringing and any visual artifacts. Its contrast-enhancement subsystem deepens inky blacks, lifts luminous highlights and sculpts depth that borders on three-dimensional realism, delivering a picture that reverberates with the same visceral intensity that opened the private press event. Beyond raw pixels, the L9300 line runs on Toshiba's new Cloud platform, closing the loop on a seamless, cloud-enabled ecosystem that guarantees uninterrupted, uncompromised 4K immersion from start to finish—bringing the night's electric anticipation full circle.**

Table 5: Sample rewrite trace produced by the **GSM - Interpreter** strategy selector with (*GPT-OSS 120B*) on Mashable News Virality Data. Each row shows an excerpt of the rewritten document, the five-category strategy combination recommended by the model (one adjective per category), and the reward score Δ at that iteration.