Canadian AI 2026
25—29 May 2026, Vancouver, BC

Stabilizing Black-Box Prompt Optimization with Textual Regularization and Signal Aggregation

MohammadReza Davari[†‡◇*], Utkarsh Garg[◇], Weixin Cai[◇], Eugene Belilovsky[†‡]
[†] Concordia University [‡] Mila – Quebec AI Institute [◇] Microsoft

Abstract

An increasing number of NLP applications interact with large language models (LLMs) through black-box APIs, making prompt engineering critical for controlling model behavior. Recent Automatic Prompt Optimization (APO) methods iteratively refine prompts using model-generated critiques (often called as *textual gradients*), but they predominantly optimize from failures and underutilize information contained in correct predictions, leading to instability and semantic drift. We propose **TRAS** (Textual Regularization with Aggregated Signals), a feedback-centric framework that is plug-and-play with existing APO search backbones. It retains the standard textual gradient signal from prior work for error correction, and introduces a complementary *textual regularizer* derived from successful predictions to preserve beneficial prompt components. Because both signals are stochastic and can be noisy, we further introduce *Monte Carlo Signal Aggregation (MCSA)*, which samples multiple gradients or regularizers and aggregates them into a single actionable directive, emphasizing consistent, actionable advice while filtering out outliers. Motivated by rapid model churn, we also formalize *Automatic Prompt Migration (APM)*, the practical problem of adapting an expert prompt across model versions or API providers without losing critical instructions. Across standard APO and APM scenarios, our approach consistently outperforms strong baselines, yielding higher accuracy, faster convergence, and lower query cost, while substantially reducing the degradation observed under naive prompt migration.[1]

Keywords: Textual Regularization, Black-box Prompt Optimization, Prompt Migration, Large Language Models

1. Introduction

Traditionally, NLP tasks have relied on fine-tuning pretrained models [1–5] on downstream datasets [6–11], leveraging internal representations such as hidden states [12, 13], gradients, and attention patterns [14–17] to drive methods like prompt tuning [18, 19], LoRA [20], and other parameter-efficient approaches [21–24]. However, the landscape is increasingly shifting toward closed-weight LLMs accessed via black-box APIs [25, 26], where internal representations are unavailable and prompt design becomes the primary mechanism for controlling model behavior [27–33]. Manual prompt engineering remains costly, requiring domain expertise and substantial trial-and-error [34–36].

Recent *Automatic Prompt Optimization* (APO) methods [37–40] iteratively refine prompts using feedback from model behavior. In many such approaches, the dominant signal is an LLM-generated critique from incorrect predictions, a *textual gradient* [37, 40]. While effective for error correction, this view underutilizes information from successful predictions and can lead to *semantic drift*: edits that fix local failures may inadvertently degrade globally useful prompt components (Figure 1). Importantly, correct predictions carry information

[1] Code is available at https://github.com/rezazzr/TRAS.

[*] mohammadreza.davari@concordia.ca

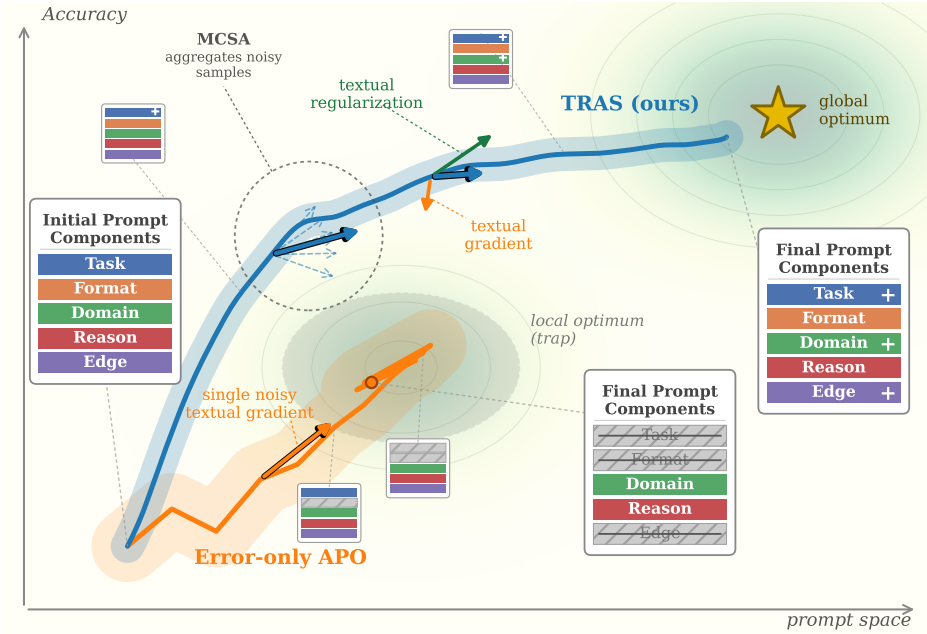


Figure 1. TRAS preserves prompt structure across optimization. Error-only APO (orange) updates prompts from failures alone; the trajectory oscillates with a wide variance band and working prompt components get erased (*instruction loss*). TRAS (blue) adds a textual regularizer from successes and aggregates multiple samples per signal (MCSA), producing a smoother trajectory with a narrower variance band that better approaches the global optimum while preserving and refining components. Illustrative of the APM regime (`GPT-3.5-turbo`→`GPT-4o`); full results in Table 3.

about *which parts of a prompt are working*, and ignoring this signal makes optimization less stable.

In this paper, we focus on the quality and reliability of the textual update signal, an axis orthogonal to the choice of search strategy [37, 40, 41] (which selects *which* prompt to refine). We introduce a complementary *textual regularization* signal derived from successes that encourages preserving beneficial prompt components, and *Monte Carlo Signal Aggregation (MCSA)* which samples multiple independent signals and aggregates them to reduce variance while managing dilution.

Prompt optimization becomes especially challenging under rapid model churn, where practitioners must migrate optimized prompts across model versions or providers. We formalize this as *Automatic Prompt Migration (APM)* [42]: adapting an expert source prompt to a target model while avoiding *instruction loss*, the inadvertent deletion of load-bearing prompt components during re-optimization [28, 39, 43–45]. This mirrors the *continual learning* setting of adapting to new tasks while preserving prior knowledge without access to past data [46], and motivates a regularization-centric update rule rather than aggressive re-optimization. Our method targets this failure mode by using textual regularization to preserve transferable structure and MCSA to reduce noisy updates.

We combine these ideas into **TRAS** (**T**extual **R**egularization with **A**ggregated **S**ignals), a unified framework for robust black-box prompt optimization and migration that consistently improves accuracy, accelerates convergence, and reduces API usage across standard APO and APM scenarios.

Our contributions are:

(1) We formalize *Automatic Prompt Migration (APM)* as a practical black-box setting, highlighting instruction loss as a key failure mode of naïve migration and re-optimization.

(2) We propose **TRAS**, augmenting textual-gradient-based APO with (i) *textual regularization* from successes to stabilize updates and (ii) *Monte Carlo Signal Aggregation* to reduce signal variance while managing dilution, yielding robust gains in both APO and APM.

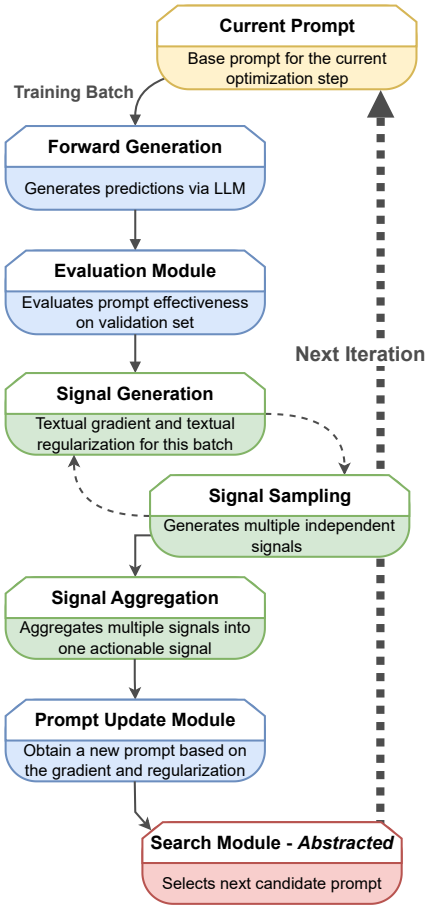


Figure 2. Overview of our proposed framework for automatic prompt optimization (APO). The framework consists of five primary modules. The Search Module is abstracted to allow for the integration of various search and planning methods.

2. Related Work

APO methods can be broadly categorized by their level of access to model internals. Methods with full or partial access to parameters, gradients, or output probabilities, applicable to open-source models like LLaMA [47–49] or Mistral [50, 51], can train soft prompts [18–20, 52, 53] or optimize discrete prompts via gradient-guided search [29, 54–58]. These techniques are infeasible for black-box APIs, our primary focus.

Black-box APO methods fall into two families. *Iterative generation* methods (e.g., APE [39], OPRO [38]) repeatedly propose prompt candidates, evaluate them on held-out data, and select top performers. These methods rely on *scalar feedback* (e.g., accuracy) and do not explicitly represent *why* a prompt fails or succeeds. *Search and planning* methods (e.g., PromptAgent [37], ProTeGi [40], SAMMO [41]) frame APO as structured exploration, using MCTS [59], beam search, or multi-objective optimization, guided by *textual gradients* derived from failures. These provide stronger exploration, but the inner-loop update signal is derived primarily from errors, which is inherently stochastic and can contribute to semantic drift. Our work complements these methods by improving the reliability and informativeness of the update signal, while remaining plug-and-play with different search strategies.

3. Method

We study APO for black-box LLMs accessed via APIs. A task is defined by $\mathcal{T} = (\mathcal{X}, \mathcal{Y}, \mathcal{D}, \ell)$, where $\mathcal{X}$ is the input space, $\mathcal{Y}$ the output space, $\mathcal{D}$ a distribution over $(x, y) \in \mathcal{X} \times \mathcal{Y}$, and ℓ a task loss. Let $\mathcal{P}$ denote the discrete prompt space; we write $p \in \mathcal{P}$ for an individual prompt. A black-box LLM induces $\hat{y} \sim P_\theta(\cdot \mid x, p)$, where we assume no access to θ, internal activations, or gradients. The APO objective is to find:

$$p^\star \in \arg\min_{p \in \mathcal{P}} R(p), \qquad R(p) := \mathbb{E}_{(x,y)\sim\mathcal{D}}\Big[\mathbb{E}_{\hat{y}\sim P_\theta(\cdot|x,p)}\big[\ell(\hat{y}, y)\big]\Big]. \tag{3.1}$$

In practice, the risk $R(p)$ is estimated on a finite validation set. Under 0–1 loss, minimizing risk is equivalent to maximizing accuracy.

Our framework is modular and can be combined with existing search backbones; it focuses on improving the *textual signals* used to revise prompts. The overall pipeline is shown in Figure 2. At each step t, given a prompt p_t, we run it on a batch, partition outcomes into successes and failures, generate corrective and stabilizing signals, aggregate them via MCSA, and rewrite the prompt. We detail each component below.

3.1. **Forward Generation and Evaluation**

At step t, we sample a mini-batch $B_t = \{(x_i, y_i)\}_{i=1}^{n}$ from the training set and obtain predictions $\hat{y}_i$ from $P_M(\cdot \mid x_i, p_t)$. We partition the batch by correctness $c_i = \mathbb{1}[\hat{y}_i = y_i]$ into an error subset B_t^- and a success subset B_t^+.

3.2. **Textual Gradient**

Following prior work [37, 40], we derive a corrective signal from incorrect predictions. A critique policy π_g produces a natural-language signal $g_t \sim \pi_g(\cdot \mid p_t, B_t^-)$ that identifies *what to change* (e.g., missing constraints, ambiguous instructions, formatting issues) to fix observed failures.

3.3. **Textual Regularization**

Error-driven updates alone can cause *semantic drift*: edits that fix local failures may delete globally beneficial prompt components, forcing the optimizer into costly *remove-and-rediscover* cycles where useful instructions are first overwritten and must later be re-learned. To stabilize optimization, we introduce a success-conditioned *textual regularization* signal that prescribes *preservation* rather than mutation. An attribution policy π_r analyzes successes and returns a regularizer $r_t \sim \pi_r(\cdot \mid p_t, B_t^+)$, which attributes correctness to specific prompt components and expresses *do-not-remove / keep-unchanged / strengthen* constraints. To avoid anchoring early iterations to a poor initialization, we employ a warm-up schedule:

$$r_t := \begin{cases} \varnothing, & t < \tau_{\text{warmup}}, \\ \text{sample from } \pi_r(\cdot \mid p_t, B_t^+), & t \geq \tau_{\text{warmup}}, \end{cases} \tag{3.2}$$

so the optimizer first explores using only corrective signals, then refines while preserving verified structure.

3.4. **Monte Carlo Signal Aggregation (MCSA)**

Both g_t and r_t are stochastic LLM outputs and can have high variance. We reduce variance by sampling K independent signals and aggregating them:

$$g_{t,1}, \ldots, g_{t,K} \overset{\text{i.i.d.}}{\sim} \pi_g(\cdot \mid p_t, B_t^-), \tag{3.3}$$

$$r_{t,1}, \ldots, r_{t,K} \overset{\text{i.i.d.}}{\sim} \pi_r(\cdot \mid p_t, B_t^+). \tag{3.4}$$

An aggregation operator (LLM summarizer) combines samples into consolidated directives:

$$\bar{g}_t := \Phi_g(\{g_{t,k}\}_{k=1}^{K}), \qquad \bar{r}_t := \Phi_r(\{r_{t,k}\}_{k=1}^{K}). \tag{3.5}$$

The aggregated signals emphasize consistent, actionable edits and de-emphasize outliers. Increasing K reduces sampling variance but can increase dilution when too many signals are compressed into a fixed-length directive, producing generic summaries. This variance–dilution trade-off yields a sweet spot at moderate K, which we characterize empirically in Section 4.3.

3.5. Prompt Update

Given p_t and aggregated signals $(\bar{g}_t, \bar{r}_t)$, we generate the next prompt via an update policy:

$$p_{t+1} \sim \pi_{\text{upd}}(\cdot \mid p_t, \bar{g}_t, \bar{r}_t, B_t), \tag{3.6}$$

which applies corrective changes from $\bar{g}_t$ while preserving components identified by $\bar{r}_t$. TRAS is orthogonal to the upstream search strategy that selects which prompt p_t to refine; in our experiments we use the MCTS strategy that was used in PromptAgent [37] as the search backbone due to its established effectiveness [60, 61].

3.6. TRAS for Automatic Prompt Migration

We formalize *prompt migration* as adapting an expert prompt $p_S^\star$, optimized for a source LLM P_{θ_S} (e.g., `GPT-3.5-turbo`), to a target LLM P_{θ_T} (e.g., `GPT-4o`):

$$\min_{p \in \mathcal{P}} R_T(p) \quad \text{s.t.} \quad p \in \mathcal{N}(p_S^\star), \tag{3.7}$$

where R_T is the target-model risk and $\mathcal{N}(\cdot)$ encodes preservation of successful prompt structure. Naïve error-driven updates can quickly overwrite load-bearing instructions in $p_S^\star$. We initialize $p_0 := p_S^\star$ and activate textual regularization *immediately* ($\tau_{\text{warmup}} := 0$), so that r_t is sampled for all $t \geq 0$. This allows TRAS to correct target-specific mismatches via $\bar{g}_t$ while preserving the transferable structure of $p_S^\star$ via $\bar{r}_t$. We also use a smaller MCSA sample count K in this regime (see Section 4.2) because the warm-started expert prompt produces lower-variance signals than the cold-start APO setting.

4. Experiments

We evaluate TRAS on five reasoning tasks spanning causal, spatial, tabular, entailment, and semantic similarity, using `GPT-3.5-turbo` (APO) and `GPT-4o` (APM on GPT-3.5-turbo to GPT-4o). We optimize prompts on validation sets and report final performance on test sets; the primary metric is accuracy. Full details on tasks, dataset statistics, splits, and default prompts are provided in Appendix A and Appendix B.

4.1. Standard Prompt Optimization (GPT-3.5-turbo)

We evaluate TRAS in the standard APO setting using `GPT-3.5-turbo`. Each run is initialized with a task-specific default prompt (Appendix B) and run for up to 15 iterations using the PromptAgent [37] search backbone. MCSA samples $K = 6$ signals per batch, and textual regularization activates after a short warm-up ($\tau_{\text{warmup}} = 3$ or 4 depending on the dataset; see Section 4.3). Table 1 reports results averaged over five seeds. TRAS yields consistent and statistically significant accuracy improvements across all tasks, ranging from 4.9% to 21.5% over PromptAgent ($p < 0.01$). MCSA reduces standard deviation across seeds, highlighting the role of aggregated signals in stabilizing optimization. TRAS also reduces LLM calls by 0.5% to 3.3% relative to the baseline by avoiding unnecessary removal-and-rediscovery cycles (Section 4.4).

4.2. Prompt Migration: GPT-3.5-turbo →GPT-4o

We evaluate TRAS in the APM setting, where expert prompts optimized on `GPT-3.5-turbo` initialize optimization on `GPT-4o`. Throughout this subsection, *DP* (Default Prompt) refers to the task-specific default prompt used as a cold-start initializer (see Appendix B), and *EP* (Expert Prompt) refers to a prompt already optimized on `GPT-3.5-turbo` that we transfer as the initializer on `GPT-4o`. As shown in Table 2, direct transfer of expert prompts yields initial gains of 1.3%–3.3% over default prompts, but naïve re-optimization via PromptAgent

Dataset (Init. Acc.)	Method	Accuracy	p-value	Cohen's d
Causal Judgment (56.5 ± 3.67)	Baseline	58.6 ± 3.98	-	-
	Baseline + MCSA*	60.8 ± 2.38	0.020	1.687
	Baseline + TR*	63.6 ± 2.62	0.040	1.336
	TRAS**	**64.4** ± 2.16	0.008	2.162
Geometric Shapes (32.7 ± 2.04)	Baseline	52.1 ± 4.94	-	-
	Baseline + MCSA*	57.8 ± 3.15	0.032	1.439
	Baseline + TR*	61.6 ± 4.18	0.035	1.399
	TRAS**	**63.3** ± 1.16	0.004	2.644
Penguins (60.5 ± 4.87)	Baseline	65.1 ± 4.96	-	-
	Baseline + MCSA*	66.1 ± 2.42	0.083	1.148
	Baseline + TR*	66.9 ± 3.97	0.043	1.464
	TRAS**	**68.6** ± 1.87	0.007	2.556
Biosses (25.2 ± 3.84)	Baseline	62.5 ± 4.19	-	-
	Baseline + MCSA*	67.0 ± 2.92	0.044	1.456
	Baseline + TR*	68.2 ± 3.02	0.021	1.844
	TRAS**	**70.4** ± 2.02	0.006	2.654
CB (68.5 ± 4.22)	Baseline	81.7 ± 3.17	-	-
	Baseline + MCSA*	84.2 ± 2.02	0.032	1.610
	Baseline + TR*	84.2 ± 3.73	0.049	1.402
	TRAS**	**85.7** ± 3.54	0.008	2.495

Table 1. Prompt optimization on `GPT-3.5-turbo` (mean accuracy ± std., five seeds). Significance vs. PromptAgent is via paired t-tests (p, Cohen's d). +MCSA: Monte Carlo Signal Aggregation (K=6); +TR: Textual Regularization; TRAS: both. Init. Acc.: default prompt. All methods run 15 iterations; +TR activates at τ_{warmup}=3 (Causal Judgment/Geometric Shapes/Penguins) or 4 (Biosses/CB). TRAS improves accuracy and reduces variance over baselines.

Dataset	Stage	DP Acc.	EP Acc.	p-value	Cohen's d
Causal Judgment	Initial	71.8 ± 1.92	**74.2** ± 3.46	0.033*	1.434
	Final	73.4 ± 1.82	**73.8** ± 1.79	0.803	0.119
Geometric Shapes	Initial	54.8 ± 1.89	**58.2** ± 2.22	0.001**	4.138
	Final	**79.0** ± 5.32	75.1 ± 5.80	0.040*	-1.344
Penguins	Initial	92.9 ± 1.85	**95.8** ± 1.72	0.025*	1.745
	Final	**95.2** ± 2.03	92.3 ± 2.89	0.005**	-2.771
Biosses	Initial	69.9 ± 2.73	**72.2** ± 1.78	0.0125*	2.159
	Final	**76.7** ± 2.84	76.1 ± 1.97	0.381	-0.600
CB	Initial	79.3 ± 1.57	**80.3** ± 1.54	0.005**	2.855
	Final	**80.0** ± 4.62	78.7 ± 2.13	0.381	-0.492

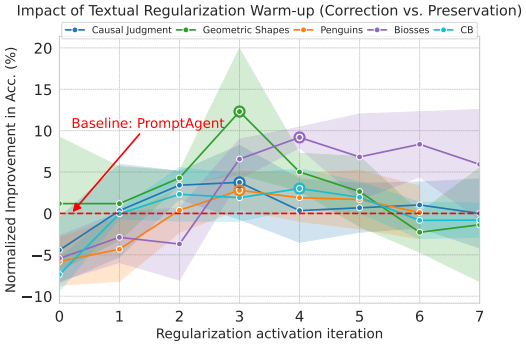
Table 2. Transferability of expert prompts (EP) from `GPT-3.5-turbo` to `GPT-4o` (accuracy mean ± std., five seeds). We compare Default Prompt (DP) vs. transferred EP at *Initial* (direct transfer) and *Final* (after 15 PromptAgent iterations on `GPT-4o`). Significance via paired t-tests. Direct transfer yields modest gains (1.3–3.3%), but naïve re-optimization with PromptAgent often erodes them (instruction loss).

often erodes these gains due to instruction loss, early corrective signals overwrite useful instructions embedded in the expert prompt.

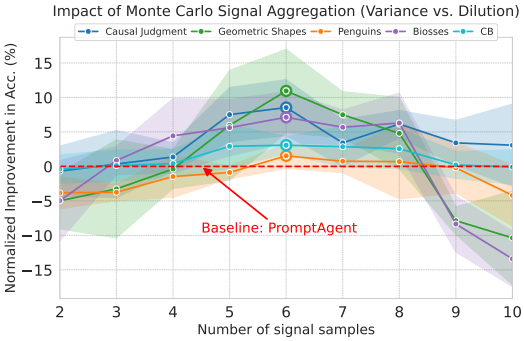
We next evaluate TRAS for APM. TRAS activates textual regularization from the first iteration ($\tau_{\text{warmup}} = 0$) and uses $K = 2$ for MCSA. Two factors motivate the smaller K: (i) the warm-started expert prompt is already strong, so per-iteration signals are less variable

Dataset (Init. Acc.)	Method	Final Acc.	p-value	Cohen's d
Causal Judgment DP: 71.8 ± 1.92 EP: 74.2 ± 3.46	Baseline	73.8 ± 1.79	-	-
	Baseline + MCSA*	74.7 ± 1.44	0.053	1.214
	Baseline + TR*	75.8 ± 1.78	0.047	1.265
	TRAS**	**76.4** ± 1.59	0.007	2.280
Geometric Shapes DP: 54.8 ± 1.89 EP: 58.2 ± 2.22	Baseline	75.1 ± 5.80	-	-
	Baseline + MCSA*	79.4 ± 2.92	0.016	1.782
	Baseline + TR*	81.7 ± 3.07	0.021	1.641
	TRAS**	**84.5** ± 2.33	0.008	2.175
Penguins DP: 92.9 ± 1.85 EP: 95.8 ± 1.72	Baseline	92.3 ± 2.89	-	-
	Baseline + MCSA*	94.2 ± 1.33	0.083	1.148
	Baseline + TR*	96.7 ± 0.88	0.041	1.493
	TRAS**	**98.0** ± 0.73	0.008	2.449
Biosses DP: 69.9 ± 2.73 EP: 72.2 ± 1.78	Baseline	76.1 ± 1.97	-	-
	Baseline + MCSA*	78.4 ± 1.66	0.043	1.461
	Baseline + TR*	83.7 ± 3.86	0.003	3.186
	TRAS**	**88.3** ± 2.00	0.0001	8.696
CB DP: 79.3 ± 1.57 EP: 80.3 ± 1.54	Baseline	78.7 ± 2.13	-	-
	Baseline + MCSA*	82.7 ± 2.31	0.029	1.659
	Baseline + TR*	85.3 ± 4.49	0.014	2.047
	TRAS**	**87.5** ± 3.56	0.006	2.713

Table 3. APM from `GPT-3.5-turbo` to `GPT-4o` (accuracy mean ± std., five seeds). We report DP/EP initial accuracy and final `GPT-4o` accuracy after optimization (all start from EP). Baseline: PromptAgent; +MCSA: aggregation (K=2); +TR: Textual Regularization from iter. 0; TRAS: both. Significance vs. baseline via paired t-tests (p, Cohen's d; $^*p < 0.05$, $^{**}p < 0.01$). TRAS achieves the best final accuracy (2.2% to 16.0% gain).



(a) Relative accuracy gain vs. correction-only baseline as a function of the textual-regularization warm-up τ_{warmup}. Performance peaks at $\tau_{\text{warmup}} \approx 3$–4; too early anchors weak prompts, too late increases drift/instruction loss.



(b) Relative accuracy gain vs. no-MCSA baseline as a function of aggregation samples K. Accuracy improves up to $K \approx 6$ (lower variance) then drops for larger K due to dilution/compression.

than in the cold-start APO setting and pilot runs showed no accuracy gain from $K \geq 3$; and (ii) `GPT-4o` API costs scale with K, so larger sweeps were not tractable at this budget. Table 3 summarizes results. TRAS consistently improves performance across all tasks, with accuracy gains of 3.5% to 16.0% over the baseline ($p < 0.01$). The largest gains occur in Geometric Shapes (12.5%) and Biosses (16.0%), where preserving domain-specific structure is especially important. TRAS also reduces total LLM calls by 4.2% to 6.2% (Section 4.4).

Model	Method	Causal Judgment	Geometric Shapes	Penguins	Biosses	CB
GPT-3.5	Baseline	7,670	11,264	2,664	3,566	5,490
	+MCSA	7,919 (+3.2%)	11,731 (+4.1%)	2,735 (+2.7%)	3,710 (+4.0%)	5,606 (+2.1%)
	+TR	7,040 (-8.2%)	10,900 (-3.2%)	2,533 (-4.9%)	3,294 (-7.6%)	5,337 (-2.8%)
	TRAS	7,429 (-3.1%)	11,204 (-0.5%)	2,622 (-1.5%)	3,449 (-3.3%)	5,422 (-1.2%)
GPT-4o	Baseline	8,576	9,642	2,980	3,008	6,229
	+MCSA	9,271 (+8.1%)	10,093 (+4.7%)	3,068 (+2.9%)	3,070 (+2.0%)	6,394 (+2.6%)
	+TR	7,963 (-7.1%)	9,396 (-2.6%)	2,909 (-2.4%)	2,869 (-4.6%)	6,079 (-2.4%)
	TRAS	8,041 (-6.2%)	9,049 (-6.1%)	2,825 (-5.2%)	2,860 (-4.9%)	5,965 (-4.2%)

Table 4. Average number of API calls for prompt optimization (APO on GPT-3.5) and migration (APM on GPT-4o), averaged over five runs. Green/red = fewer/more calls vs. baseline. +MCSA: Monte Carlo Signal Aggregation K=6 (APO) / K=2 (APM); +TR: Textual Regularization; TRAS: both.

4.3. Ablation Studies

All ablations use `GPT-3.5-turbo`, run for 8 iterations from task-specific default prompts (Appendix B), and report relative accuracy improvements over the PromptAgent baseline (dotted red line in Figures 3a and 3b).

Ablation 1: Textual regularization warm-up (τ_{warmup}) We vary τ_{warmup} from 0 to 7 (Figure 3a). Early activation ($\tau_{\text{warmup}} \in \{0, 1\}$) underperforms, consistent with premature anchoring to weak prompt structure. Peak performance occurs at $\tau_{\text{warmup}} = 3$ for Causal Judgment, Geometric Shapes, and Penguins, and at $\tau_{\text{warmup}} = 4$ for CB and Biosses. Beyond these values, performance declines as delayed preservation increases semantic drift.

Ablation 2: MCSA aggregation samples (K) We apply MCSA to the baseline using only the corrective signal (Figure 3b). Performance improves steadily up to $K \approx 6$, then degrades for larger K (e.g., 8–10) as dilution from over-compression produces generic summaries. These findings validate the variance–dilution trade-off and motivate $K = 6$ in our main APO experiments.

4.4. Experimentation Costs

Table 4 reports the average number of API calls required by each method during prompt optimization (APO) and migration (APM) for both `GPT-3.5-turbo` and `GPT-4o`, averaged over five seeds per task. Methods employing MCSA incur additional calls per iteration due to multiple signal samples, but improved signal quality reduces the number of iterations needed and avoids inefficient "remove-and-rediscover" cycles. In APO, TRAS often reduces total API calls relative to PromptAgent (our default search backbone) by converging more reliably. In APM, where instruction loss is a primary failure mode, efficiency gains are more pronounced: TRAS reduces total calls by 4.2%–6.2% while also improving accuracy.

5. Conclusion and Future Work

We presented **TRAS**, a feedback-centric framework that stabilizes black-box APO by augmenting standard textual gradients with *textual regularization* from successes and *Monte Carlo Signal Aggregation* to reduce signal variance. We also formalized *Automatic Prompt Migration (APM)* as adapting expert prompts across model distributions while avoiding instruction loss. Across both APO and APM scenarios, TRAS consistently improves accuracy, reduces variance, accelerates convergence, and lowers API usage. Promising future

directions include alternative aggregation mechanisms (e.g., structured voting or embedding-space clustering), adaptive regularization schedules, and extension to multi-turn or continual deployment settings.

Acknowledgements

We acknowledge funding from the NSERC Discovery Grant RGPIN-2021-04104 and FRQNT New Scholar. This research was enabled in part by compute resources provided by Digital Research Alliance of Canada (the Alliance) and Calcul Québec.

References

[1] R. Bommasani, D. A. Hudson, E. Adeli, R. Altman, S. Arora, S. von Arx, M. S. Bernstein, J. Bohg, A. Bosselut, E. Brunskill, et al. "On the opportunities and risks of foundation models". In: *arXiv preprint arXiv:2108.07258* (2021).

[2] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova. "Bert: Pre-training of deep bidirectional transformers for language understanding". In: *Proceedings of the 2019 conference of the North American chapter of the association for computational linguistics: human language technologies, volume 1 (long and short papers)*. 2019, pp. 4171–4186.

[3] M. Lewis, Y. Liu, N. Goyal, M. Ghazvininejad, A. Mohamed, O. Levy, V. Stoyanov, and L. Zettlemoyer. "BART: Denoising Sequence-to-Sequence Pre-training for Natural Language Generation, Translation, and Comprehension". In: *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics, July 2020, pp. 7871–7880.

[4] A. Radford, K. Narasimhan, T. Salimans, I. Sutskever, et al. "Improving language understanding by generative pre-training". In: (2018).

[5] C. Raffel, N. Shazeer, A. Roberts, K. Lee, S. Narang, M. Matena, Y. Zhou, W. Li, and P. J. Liu. "Exploring the limits of transfer learning with a unified text-to-text transformer". In: *Journal of machine learning research* 21.140 (2020), pp. 1–67.

[6] M. Davari, L. Kosseim, and T. D. Bui. "Toponym identification in epidemiology articles–a deep learning approach". In: *International Conference on Computational Linguistics and Intelligent Text Processing*. Springer. 2019, pp. 26–37.

[7] F. Farahnak, E. Mohammadi, M. Davari, and L. Kosseim. "Semantic Similarity Matching Using Contextualized Representations." In: *Canadian AI*. 2021.

[8] M. Davari, L. Kosseim, and T. Bui. "TIMBERT: toponym identifier for the medical domain based on BERT". In: *Proceedings of the 28th International Conference on Computational Linguistics*. 2020, pp. 662–668.

[9] Z. Yang, Y. Maricar, M. Davari, N. Grenon-Godbout, and R. Rabbany. "Toxbuster: In-game chat toxicity buster with BERT". In: *arXiv preprint arXiv:2305.12542* (2023).

[10] J. Marks, M. Davari, and L. Kosseim. "Clac at semeval-2024 task 2: Faithful clinical trial inference". In: *Proceedings of the 18th International Workshop on Semantic Evaluation (SemEval-2024)*. 2024, pp. 1673–1677.

[11] M. Davari. "Neural network approaches to medical toponym recognition". PhD thesis. Concordia University, 2020.

[12] A. Rogers, O. Kovaleva, and A. Rumshisky. "A primer in BERTology: What we know about how BERT works". In: *Transactions of the association for computational linguistics* 8 (2021), pp. 842–866.

[13] M. Davari, N. Asadi, S. Mudur, R. Aljundi, and E. Belilovsky. "Probing representation forgetting in supervised and unsupervised continual learning". In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2022, pp. 16712–16721.

[14] S. Kornblith, M. Norouzi, H. Lee, and G. Hinton. "Similarity of neural network representations revisited". In: *International conference on machine learning*. PMLR. 2019, pp. 3519–3529.

[15] M. Raghu, T. Unterthiner, S. Kornblith, C. Zhang, and A. Dosovitskiy. "Do vision transformers see like convolutional neural networks?" In: *Advances in neural information processing systems* 34 (2021), pp. 12116–12128.

[16] M. Davari, S. Horoi, A. Natik, G. Lajoie, G. Wolf, and E. Belilovsky. "Reliability of CKA as a Similarity Measure in Deep Learning". In: *The Eleventh International Conference on Learning Representations*. 2023.

[17] M. Davari, S. Horoi, A. Natik, G. Lajoie, G. Wolf, and E. Belilovsky. "On the inadequacy of CKA as a measure of similarity in deep learning". In: *ICLR 2022 Workshop on Geometrical and Topological Representation Learning*. 2022.

[18] X. L. Li and P. Liang. "Prefix-tuning: Optimizing continuous prompts for generation". In: *arXiv preprint arXiv:2101.00190* (2021).

[19] B. Lester, R. Al-Rfou, and N. Constant. "The power of scale for parameter-efficient prompt tuning". In: *arXiv preprint arXiv:2104.08691* (2021).

[20] E. J. Hu, Y. Shen, P. Wallis, Z. Allen-Zhu, Y. Li, S. Wang, L. Wang, W. Chen, et al. "Lora: Low-rank adaptation of large language models." In: *ICLR* 1.2 (2022), p. 3.

[21] M. Davari and E. Belilovsky. "Model breadcrumbs: scalable upcycling of finetuned foundation models via sparse task vectors merging". In: *ICML 2024 Workshop on Foundation Models in the Wild*. 2024.

[22] P. Yadav, D. Tam, L. Choshen, C. A. Raffel, and M. Bansal. "Ties-merging: Resolving interference when merging models". In: *Advances in Neural Information Processing Systems* 36 (2023), pp. 7093–7115.

[23] M. Davari and E. Belilovsky. "Model breadcrumbs: Scaling multi-task model merging with sparse masks". In: *European Conference on Computer Vision*. Springer. 2024, pp. 270–287.

[24] L. Yu, B. Yu, H. Yu, F. Huang, and Y. Li. "Language models are super mario: Absorbing abilities from homologous models as a free lunch". In: *Forty-first International Conference on Machine Learning*. 2024.

[25] OpenAI. "Gpt-4 technical report". In: *arXiv preprint arXiv:2303.08774* (2023).

[26] S. Bubeck, V. Chadrasekaran, R. Eldan, J. Gehrke, E. Horvitz, E. Kamar, P. Lee, Y. T. Lee, Y. Li, S. Lundberg, et al. *Sparks of artificial general intelligence: Early experiments with gpt-4*. 2023.

[27] R. Luo, L. Sun, Y. Xia, T. Qin, S. Zhang, H. Poon, and T.-Y. Liu. "BioGPT: generative pre-trained transformer for biomedical text generation and mining". In: *Briefings in bioinformatics* 23.6 (2022), bbac409.

[28] T. Kojima, S. S. Gu, M. Reid, Y. Matsuo, and Y. Iwasawa. "Large language models are zero-shot reasoners". In: *Advances in neural information processing systems* 35 (2022), pp. 22199–22213.

[29] X. Wang, J. Wei, D. Schuurmans, Q. Le, E. Chi, S. Narang, A. Chowdhery, and D. Zhou. "Self-consistency improves chain of thought reasoning in language models". In: *arXiv preprint arXiv:2203.11171* (2022).

[30] D. Zhou, N. Schärli, L. Hou, J. Wei, N. Scales, X. Wang, D. Schuurmans, C. Cui, O. Bousquet, Q. Le, et al. "Least-to-most prompting enables complex reasoning in large language models". In: *arXiv preprint arXiv:2205.10625* (2022).

[31] A. Madaan, N. Tandon, P. Gupta, S. Hallinan, L. Gao, S. Wiegreffe, U. Alon, N. Dziri, S. Prabhumoye, Y. Yang, et al. "Self-refine: Iterative refinement with self-feedback". In: *Advances in Neural Information Processing Systems* 36 (2023), pp. 46534–46594.

[32] Y. Bai, S. Kadavath, S. Kundu, A. Askell, J. Kernion, A. Jones, A. Chen, A. Goldie, A. Mirhoseini, C. McKinnon, et al. "Constitutional ai: Harmlessness from ai feedback". In: *arXiv preprint arXiv:2212.08073* (2022).

[33] X. Chen, M. Lin, N. Schärli, and D. Zhou. "Teaching large language models to self-debug". In: *arXiv preprint arXiv:2304.05128* (2023).

[34] J. Jiang, K. Zhou, Z. Dong, K. Ye, W. X. Zhao, and J.-R. Wen. "Structgpt: A general framework for large language model to reason over structured data". In: *arXiv preprint arXiv:2305.09645* (2023).

[35] J. Wei, X. Wang, D. Schuurmans, M. Bosma, F. Xia, E. Chi, Q. V. Le, D. Zhou, et al. "Chain-of-thought prompting elicits reasoning in large language models". In: *Advances in neural information processing systems* 35 (2022), pp. 24824–24837.

[36] A. Kong, S. Zhao, H. Chen, Q. Li, Y. Qin, R. Sun, X. Zhou, E. Wang, and X. Dong. "Better zero-shot reasoning with role-play prompting". In: *arXiv preprint arXiv:2308.07702* (2023).

[37] X. Wang, C. Li, Z. Wang, F. Bai, H. Luo, J. Zhang, N. Jojic, E. P. Xing, and Z. Hu. "Promptagent: Strategic planning with language models enables expert-level prompt optimization". In: *arXiv preprint arXiv:2310.16427* (2023).

[38] C. Yang, X. Wang, Y. Lu, H. Liu, Q. V. Le, D. Zhou, and X. Chen. "Large language models as optimizers". In: *arXiv preprint arXiv:2309.03409* (2023).

[39] Y. Zhou, A. I. Muresanu, Z. Han, K. Paster, S. Pitis, H. Chan, and J. Ba. "Large language models are human-level prompt engineers". In: *arXiv preprint arXiv:2211.01910* (2022).

[40] R. Pryzant, D. Iter, J. Li, Y. T. Lee, C. Zhu, and M. Zeng. "Automatic prompt optimization with" gradient descent" and beam search". In: *arXiv preprint arXiv:2305.03495* (2023).

[41] T. Schnabel and J. Neville. "Symbolic prompt program search: A structure-aware approach to efficient compile-time prompt optimization". In: *arXiv preprint arXiv:2404.02319* (2024).

[42] M. Davari. "Continual Learning in Constrained Scenarios: Bridging Real-World Needs and Practical Constraints". Unpublished. PhD thesis. Concordia University, 2025. URL: https://spectrum.library.concordia.ca/id/eprint/995684/.

[43] T. Zhang, X. Wang, D. Zhou, D. Schuurmans, and J. E. Gonzalez. "Tempera: Test-time prompting via reinforcement learning". In: *arXiv preprint arXiv:2211.11890* (2022).

[44] X. Ma, S. Mishra, A. Beirami, A. Beutel, and J. Chen. "Let's Do a Thought Experiment: Using Counterfactuals to Improve Moral Reasoning". In: *arXiv preprint arXiv:2306.14308* (2023).

[45] J. Chen, L. Chen, H. Huang, and T. Zhou. "When do you need Chain-of-Thought Prompting for ChatGPT?" In: *arXiv preprint arXiv:2304.03262* (2023).

[46] N. Asadi, M. Davari, S. Mudur, R. Aljundi, and E. Belilovsky. "Prototype-Sample Relation Distillation: Towards Replay-Free Continual Learning". In: *International Conference on Machine Learning*. PMLR. 2023, pp. 1093–1106.

[47] H. Touvron, T. Lavril, G. Izacard, X. Martinet, M.-A. Lachaux, T. Lacroix, B. Rozière, N. Goyal, E. Hambro, F. Azhar, et al. "Llama: Open and efficient foundation language models". In: *arXiv preprint arXiv:2302.13971* (2023).

[48] H. Touvron, L. Martin, K. Stone, P. Albert, A. Almahairi, Y. Babaei, N. Bashlykov, S. Batra, P. Bhargava, S. Bhosale, et al. "Llama 2: Open foundation and fine-tuned chat models". In: *arXiv preprint arXiv:2307.09288* (2023).

[49] A. Grattafiori, A. Dubey, A. Jauhri, A. Pandey, A. Kadian, A. Al-Dahle, A. Letman, A. Mathur, A. Schelten, A. Vaughan, et al. "The llama 3 herd of models". In: *arXiv preprint arXiv:2407.21783* (2024).

[50] A. Q. Jiang, A Sablayrolles, A Mensch, C Bamford, D. S. Chaplot, D. Casas, F Bressand, G Lengyel, G Lample, L Saulnier, et al. "Mistral 7b. arxiv". In: *arXiv preprint arXiv:2310.06825* 10 (2023).

[51] M. A. team. *Mistral nemo*. Accessed: 2025. 2024. URL: https://mistral.ai/news/mistral-nemo.

[52] Z. Wang, R. Panda, L. Karlinsky, R. Feris, H. Sun, and Y. Kim. "Multitask prompt tuning enables parameter-efficient transfer learning". In: *arXiv preprint arXiv:2303.02861* (2023).

[53] G. Qin and J. Eisner. "Learning how to ask: Querying LMs with mixtures of soft prompts". In: *arXiv preprint arXiv:2104.06599* (2021).

[54] T. Shin, Y. Razeghi, R. L. Logan IV, E. Wallace, and S. Singh. "Autoprompt: Eliciting knowledge from language models with automatically generated prompts". In: *arXiv preprint arXiv:2010.15980* (2020).

[55] Y. Wen, N. Jain, J. Kirchenbauer, M. Goldblum, J. Geiping, and T. Goldstein. "Hard prompts made easy: Gradient-based discrete optimization for prompt tuning and discovery". In: *Advances in Neural Information Processing Systems* 36 (2023), pp. 51008–51025.

[56] T. Gao, A. Fisch, and D. Chen. "Making pre-trained language models better few-shot learners". In: *arXiv preprint arXiv:2012.15723* (2020).

[57] L. Chen, J. Chen, T. Goldstein, H. Huang, and T. Zhou. "Instructzero: Efficient instruction optimization for black-box large language models". In: *arXiv preprint arXiv:2306.03082* (2023).

[58] Y. Hao, Z. Chi, L. Dong, and F. Wei. "Optimizing prompts for text-to-image generation". In: *Advances in Neural Information Processing Systems* 36 (2023), pp. 66923–66939.

[59] R. Coulom. “Efficient selectivity and backup operators in Monte-Carlo tree search”. In: *International conference on computers and games*. Springer. 2006, pp. 72–83.

[60] J. Zhang, Z. Wang, H. Zhu, J. Liu, Q. Lin, and E. Cambria. “MARS: A Multi-Agent Framework Incorporating Socratic Guidance for Automated Prompt Optimization”. In: *arXiv preprint arXiv:2503.16874* (2025).

[61] W. Li, X. Wang, W. Li, and B. Jin. “A Survey of Automatic Prompt Engineering: An Optimization Perspective”. In: *arXiv preprint arXiv:2502.11560* (2025).

[62] M. Suzgun, N. Scales, N. Schärli, S. Gehrmann, Y. Tay, H. W. Chung, A. Chowdhery, Q. V. Le, E. H. Chi, D. Zhou, et al. “Challenging big-bench tasks and whether chain-of-thought can solve them”. In: *arXiv preprint arXiv:2210.09261* (2022).

[63] M.-C. De Marneffe, M. Simons, and J. Tonhauser. “The commitmentbank: Investigating projection in naturally occurring discourse”. In: *proceedings of Sinn und Bedeutung*. Vol. 23. 2019, pp. 107–124.

[64] G. Soğancıoğlu, H. Öztürk, and A. Özgür. “BIOSSES: a semantic sentence similarity estimation system for the biomedical domain”. In: *Bioinformatics* 33.14 (2017), pp. i49–i58.

[65] A. Wang, Y. Pruksachatkun, N. Nangia, A. Singh, J. Michael, F. Hill, O. Levy, and S. Bowman. “Superglue: A stickier benchmark for general-purpose language understanding systems”. In: *Advances in neural information processing systems* 32 (2019).

Dataset	Train	Validation	Test
Causal Judgment	30	60	100
Geometric Shapes	50	100	200
Penguins	30	40	79
Biosses	30	30	40
CB	30	95	56

Table 5. Dataset splits used for prompt optimization and evaluation.

Appendix A. Dataset

We evaluate TRAS on five tasks spanning causal, spatial, tabular, inferential, and semantic reasoning, covering both classification and regression settings. Three tasks are drawn from BBH [62], a widely used benchmark in prompt optimization [37–41]: **Causal Judgment** (binary causal inference), **Geometric Shapes** (multi-class reasoning over SVG path strings), and **Penguins** (table-based classification over structured data). We additionally include two non-BBH benchmarks: **CommitmentBank (CB)** [63], a natural language inference dataset (entailment/contradiction/neutral), and **Biosses** [64], a biomedical semantic similarity benchmark (regression over continuous similarity scores). Table 5 summarizes the dataset splits.

Causal Judgment This dataset tests causal attribution skills by presenting real-world scenarios and asking whether one event caused another. It evaluates the model's ability to perform commonsense reasoning under ambiguity. Below is an example instance from the dataset:

```
Joe was feeling quite dehydrated, so he stopped by the local smoothie shop to buy
the largest sized drink available.  Before ordering, the cashier told him that the
Mega-Sized Smoothies were now one dollar more than they used to be.  Joe replied, "I
don't care if I have to pay one dollar more, I just want the biggest smoothie you
have."  Sure enough, Joe received the Mega-Sized Smoothie and paid one dollar more for
it.  Did Joe intentionally pay one dollar more?
Label:  Yes
```

Geometric Shapes A synthetic visual-reasoning-inspired dataset that describes a geometric shape via its SVG representation. The task tests the model's ability to infer spatial and comparative relationships. An example is shown below:

```
This SVG path element <path d="M 59.43,52.76 L 75.49,27.45 L 54.92,4.40 M 54.92,4.40 L
23.70,7.77 L 15.15,42.15 L 34.51,57.44 L 59.43,52.76"/> draws a
Label:  hexagon
```

Penguins This dataset examinsines the model's ability to reason about tabular data. At each instance, the model is presented with a question about penguins in a table format, and it must select the correct answer from a set of choices.

```
Here is a table where the first line is a header and each subsequent line is a penguin:

name, age, height (cm), weight (kg)
Louis, 7, 50, 11
Bernard, 5, 80, 13
Vincent, 9, 60, 11
Gwen, 8, 70, 15
For example:  the age of Louis is 7, the weight of Gwen is 15 kg, the height of Bernard
is 80 cm.
Question:  What is the height of Gwen?
```

```
Options:  A) 50 cm, B) 80 cm, C) 70 cm, D) 60 cm

Label:  C) 70 cm
```

Biosses This biomedical sentence similarity dataset presents pairs of scientific statements and asks for semantic similarity on a scale, assessing the model's ability to reason about specialized, domain-specific language. A sample pair is shown below:

```
S1:  It has recently been shown that Craf is essential for Kras G12D-induced NSCLC.
S2:  It has recently become evident that Craf is essential for the onset of Kras-driven
non-small cell lung cancer.

Label:  Similar
```

CB (CommitmentBank) CB is a SuperGLUE [65] NLI dataset designed to evaluate pragmatic inference and speaker commitment in naturally occurring sentences. It differs from standard NLI datasets because each hypothesis is derived directly from the premise's embedded clause, minimizing annotation artifacts. Below is a representative example:

```
Premise:  Some of them, like for instance the farm in Connecticut, are quite small.  If
I like a place I buy it.  I guess you could say it's a hobby."
Hypothesis:  Buying places is a hobby.

Label:  Entailment
```

In this case, the hypothesis is the complement of the clause-embedding verb `say`, and models must correctly infer that the sentence author is committed to the embedded proposition. This task hinges on understanding modality, clause embedding, and speaker stance, rather than surface-level lexical overlap.

Appendix B. Task Prompts

Each task begins with a minimalist *base prompt* that serves as the initialization point for prompt optimization. These prompts are written as system messages in the `GPT-3.5-turbo` and `GPT-4o` chat interfaces, and are intentionally kept simple to avoid embedding task-specific strategies or heuristics. The goal is to provide just enough instruction for the model to attempt the task, allowing the optimization process to refine and expand the prompt effectively. Below, we list the base prompts used for each task.

Causal Judgment

```
Answer questions about causal attribution
```

Geometric Shapes

```
Name geometric shapes from their SVG paths
```

Penguins

```
Answer questions about a table of penguins and their attributes
```

Biosses

```
Decide if these two sentences are (A) Not similar (B) Somewhat similar (C) Similar.
```

CB

```
What is the relationship between the following premise and the hypothesis?
Options:
- Contradiction
```

```
- Neutral
- Entailment
```

In Section 4.1, we described how each base prompt is optimized. Below, we include the resulting **expert prompts** obtained from the final iteration of the standard prompt optimization process. These reflect the outcome of the optimization process when targeting performance on the `GPT-3.5` model.

Causal Judgment

```
Provide causal attributions in complex scenarios by guiding the model to thoroughly
analyze the critical steps, individual intentions, and specific actions that lead to
outcomes. Emphasize the importance of identifying and prioritizing the primary cause
in each scenario, focusing on direct causes rather than incidental factors. Define
clear criteria for evaluating factors and determining the primary cause, considering
the combined impact of multiple factors working in conjunction. Instruct the model to
weigh the influence of various factors and explicitly guide it on handling conflicting
actions and scenarios involving multiple individuals. Ensure that the model carefully
considers all significant actions, intentions, and sequences of events leading to the
final outcome to accurately attribute causation. Provide explicit instructions for
distinguishing between direct causes and incidental factors, prioritizing immediate
actions that directly influence outcomes. Define specific criteria for evaluating
factors and determining the primary cause, especially in scenarios involving multiple
individuals. Emphasize the need to analyze critical steps and actions leading to
outcomes in order to accurately attribute causation.
```

Geometric Shapes

```
Name the geometric shape accurately based on the provided SVG path. Carefully analyze
the properties of the path, including the number of sides, angles, lengths of sides,
and overall configuration, to determine the most appropriate geometric shape. Your
options should encompass a wide variety of shapes, ranging from simple polygons to
circles. Ensure that the model considers all relevant attributes before selecting the
most suitable shape from the available options.
Options: (A) circle, (B) equilateral triangle, (C) regular hexagon, (D) rhombus, (E)
line segment, (F) octagon, (G) pentagon, (H) rectangle, (I) sector, (J) square, (K)
trapezoid, (L) oval
```

Penguins

```
Answer questions regarding the following tables of penguins and giraffes, ensuring to
accurately reflect any changes made to the penguin table throughout our discussion.
Please note these modifications specifically when determining key attributes such as
age, weight, or when making comparisons between penguins and giraffes.

Penguin Table:
name, age, height (cm), weight (kg)
Louis, 7, 50, 11
Vincent, 9, 60, 11
Gwen, 8, 70, 15
(Any additions or deletions of penguins will be noted in subsequent questions)

Giraffe Table:
name, age, height (cm), weight (kg)
Jody, 5, 430, 620
Gladys, 10, 420, 590
Marian, 2, 310, 410
Donna, 9, 440, 650

For each question, provide clear and logical reasoning behind your answer. Remember
to validate the latest state of the penguin table before responding, especially when
```

involving comparisons with giraffes or assessing the attributes of the penguins.

Additionally, if modifications were made to the penguin table, please annotate them clearly in your response. This ensures that we maintain an accurate understanding of the current data.

Biosses

Decide if these two sentences are (A) Not similar (B) Somewhat similar (C) Similar. Compare the specific regulatory mechanisms and molecular pathways mentioned in each sentence to determine their similarity, explicitly identifying the role of miRNA expression and binding, as well as the relevance of the molecular characteristics of GEFs and nucleotide-binding pockets in the context of the sentences. Analyze both the similarities and differences between the sentences, focusing on the nuances of the regulatory mechanisms and molecular pathways mentioned, and considering the implications for cancer types and cellular processes

CB

What is the relationship between the following premise and the hypothesis? Premise: As the storm raged outside, with thunder clapping and lightning illuminating the dark sky, Sarah felt a wave of panic wash over her. She could hear the wind howling, and every crash of thunder made her heart race faster. Despite being tucked away under her thick blankets, she couldn't shake the feeling of terror that gripped her. The flickering candle nearby offered little comfort as she lay wide awake, listening to the chaos around her.
Hypothesis: Sarah felt a strong fear of the storm.
Entailment: The hypothesis is entailed by the premise. Sarah's panic and terror at the storm directly imply that she felt a strong fear of it. What is the relationship between the following premise and the hypothesis?
Options:
- Contradiction
- Neutral
- Entailment