# TEXTUAL GRADIENTS HAVE SCOPE: RETHINKING FEDERATED TEXTUAL OPTIMIZATION

**Anonymous authors**
Paper under double-blind review

## ABSTRACT

Federated textual optimization enables clients to improve prompts locally and aggregate natural-language updates without centrally pooling data. Existing approaches, however, typically compress distributed textual updates into a single prompt shared by all clients, implicitly treating useful textual gradients as broadly applicable. We show that this assumption often fails under heterogeneous client distributions. Across 2,070 textual-gradient rules from 108 federated runs over six language models, only $0.4\%$ are empirically classified as globally beneficial under the default validation protocol, while $85.4\%$ exhibit partial client scope. Motivated by this observation, we introduce *textual-gradient scope*, which captures the client-wise support of rule-level utility, and propose SCOPED-FEDTEXTGRAD, a scope-preserving framework for federated textual optimization. SCOPED-FEDTEXTGRAD atomizes local prompt revisions into independently routable rules, measures their target-client utility, and composes only scope-compatible updates under a hard prompt-token budget. A validation-gated variant, SAFE-SCOPED, further falls back to the local solution when scoped sharing is not supported by validation evidence. SCOPED-FEDTEXTGRAD improves average accuracy over Local TextGrad, while using approximately $73\%$ fewer prompt tokens than unconstrained concatenation. The gains persist across model scales and families, and validation gating reduces negative transfer by more than half. These results suggest that federated textual gradients should be shared according to where they are useful, rather than compressed into a universally shared description. Anonymous code is available at https://anonymous.4open.science/r/Scope-FedTextual

## 1 INTRODUCTION

Large language models (LLMs) are increasingly used not only as predictors, but also as optimizers that improve prompts, programs, and compound AI systems through natural-language feedback (Yuksekgonul et al., 2025; Pryzant et al., 2023; Yang et al., 2024). A central abstraction behind this paradigm is the *textual gradient*: an LLM-generated critique that describes how a textual variable should be revised to improve an objective. Unlike numerical gradients, textual gradients can optimize black-box components without requiring access to model parameters or differentiable computation graphs. Recent work extends this idea to federated learning, where clients optimize prompts on local data and a server aggregates the resulting textual updates into a shared global prompt (Chen et al., 2025). This provides a natural interface for collaborative textual optimization without centrally pooling client data.

Federated textual optimization, however, may involve clients with different tasks, input distributions, and local objectives. Existing methods concatenate or summarize locally optimized prompts and then broadcast the resulting prompt to every client. Concatenation preserves more local information but grows increasingly long, whereas Summary and UID (Chen et al., 2025) compress the updates into shorter common descriptions. Viewed through the lens of rate–distortion theory (Nagle et al., 2024), prompt length acts as an operational rate and each client's downstream loss defines a semantic distortion. Existing aggregation therefore constructs one rate-constrained description that is decoded under multiple client-specific distortion functions. Although a common prompt can work well, this formulation leaves one fundamental question implicit.



Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

*Which textual update should be shared with which client?*

We answer this question through the *scope* of a textual gradient. We first atomize local prompt updates into actionable rules and measure how each rule changes the validation distortion of each target client. The clients for which a rule yields positive validation utility define its *scope*. Scope is not an intrinsic semantic property of a rule: it depends on the current prompt, model, target-client objective, and data distribution. A textual gradient therefore specifies not only **what should change**, but also **where that change is currently beneficial**.

A systematic empirical analysis shows that this missing client dimension is substantial. Across $2{,}070$ textual-gradient rules collected from $108$ federated runs over six language models, under the default validation protocol, only $0.4\%$ are empirically classified as improving every client, whereas $85.4\%$ exhibit partial client scope. Textual gradients are therefore typically transferable without being universal. UID regulates how information is retained within a common description, whereas scope captures an orthogonal structure: how the utility of each update varies across clients. A common aggregator may encode such conditionality implicitly, but does not represent client–gradient applicability as an explicit optimization variable.

Motivated by this finding, we propose SCOPED-FEDTEXTGRAD, a scope-preserving framework for federated textual optimization. The method retains the original local TextGrad optimizer and changes only how local updates are represented and shared. First, each accepted local prompt revision is atomized into a small set of portable textual-gradient rules. Second, target clients measure the validation utility of each candidate rule, yielding a client–gradient utility matrix and its corresponding scope. Third, each client constructs a compact prompt by selecting only scope-compatible rules under a hard token budget. This realizes client-conditioned compression of the shared rule pool while naturally retaining common sharing when rules are broadly beneficial. Finally, SAFE-SCOPED compares the complete scoped prompt with the independently optimized local prompt and falls back when scoped sharing is not supported by local validation evidence.

Our experiments demonstrate substantial gains in both predictive quality and prompt efficiency. On Qwen3-8B, SCOPED-FEDTEXTGRAD improves average accuracy over Local TextGrad by $5.9$ percentage points under Task Skew and $7.2$ points under Compound Skew, while outperforming UID by up to $22.8$ points. It simultaneously uses approximately $73\%$ fewer prompt tokens than unconstrained concatenation. The validation-gated variant reduces the observed negative-transfer rate by more than half. Across Qwen3 models from 4B to 32B and the Llama and Phi families, the gated framework remains effective.

**Contributions.** Our contributions are as follows:

- We identify and systematically characterize *textual-gradient scope*. Across $2{,}070$ rules, $108$ federated runs, and six language models, textual gradients predominantly exhibit client-dependent rather than uniformly global utility and predominantly exhibit transferable but non-global applicability.
- We formulate federated textual optimization through an explicit client–gradient utility matrix, separating *what* a textual update proposes from *where* it is useful. This client axis complements the prompt-rate axis of rate–distortion and makes rule applicability an explicit optimization object.
- We instantiate this principle in SCOPED-FEDTEXTGRAD, combining textual-gradient atomization, client-side utility measurement, and rate-constrained scope-preserving composition, together with a validation-gated variant.
- Extensive experiments across client distributions, token budgets, tasks, model families, and model scales demonstrate improved accuracy–rate trade-offs and reduced negative transfer.

## 2 RELATED WORK

**Federated learning with foundation models and LLMs.** Classical federated learning aggregates numerical client updates, while proximal and personalized variants address statistical heterogeneity

through stabilized local training or client-specific solutions (McMahan et al., 2017; Li et al., 2020; 2021). With foundation models, the exchanged object has become increasingly lightweight and structured. Federated LLM adaptation includes low-rank or communication-efficient model tuning, exemplified by FRLoRA and Ferret (Yan et al., 2025; Shu et al., 2025), as well as prompt-based adaptation of frozen language and vision–language backbones (Zhao et al., 2023; Guo et al., 2024; Yang et al., 2023; Li et al., 2024; Luo et al., 2025; Fang et al., 2025; Qi et al., 2025; Bo et al., 2025). These methods communicate model parameters or continuous prompt vectors and handle heterogeneity through global–local decompositions, personalized prompt generators, or prompt experts. In black-box LLM settings, FedOne optimizes discrete prompts under query constraints, whereas FedTextGrad exchanges natural-language prompt updates and aggregates them through concatenation, summarization, or UID-guided summarization (Wang et al., 2025; Chen et al., 2025). Our work preserves FedTextGrad's local textual-optimization protocol but recasts federation as client-conditioned prompt compression. Unlike generic personalized federated methods, which directly construct a client-specific model or prompt, our object of interest is update-centric: we explicitly estimate the client support of each reusable natural-language update before composition.

**Automatic prompt and textual optimization.** Automatic prompt optimization treats natural-language instructions as variables in a black-box search problem. OPRO conditions an LLM on previous candidates and their scores to propose improved prompts (Yang et al., 2024), while ProTeGi generates natural-language critiques and edits prompts in the opposite semantic direction using beam search and bandit selection (Pryzant et al., 2023). PromptBreeder evolves prompt populations, and DSPy compiles declarative LM programs by optimizing their prompt components (Fernando et al., 2024; Khattab et al., 2024). TextGrad generalizes this line by representing language-model feedback as textual gradients that can be propagated through prompts and other text-valued components (Yuksekgonul et al., 2025). Recent work studies complementary properties of language feedback: REVOLVE tracks response evolution across iterations, GEPA reflects over execution trajectories and evolves reusable rules, Text2Grad maps critiques to span-level parameter updates, and textual equilibrium propagation addresses feedback degradation in deep compound systems (Zhang et al., 2025; Agrawal et al., 2026; Wang et al., 2026; Chen et al., 2026). Prompt-compression work further formalizes the trade-off between prompt length and downstream distortion, emphasizing the value of conditioning compression on the downstream query or objective (Nagle et al., 2024). These studies primarily improve how textual updates are generated, propagated, assigned credit, or compressed for a downstream objective. We study the federated case in which the same textual rule is evaluated under multiple client-specific distortion functions, making its target-dependent scope an explicit optimization object.

## 3 Textual Gradients Have Scope

Consider a federation of clients $\mathcal{C} = \{1, \dots, K\}$, where client $i$ holds local data drawn from $\mathcal{P}_i$. We allow $\mathcal{P}_i \neq \mathcal{P}_j$, reflecting differences in tasks, input difficulty, or local objectives. All clients share a prompt-controlled language model $f_\theta$ and begin a communication round from prompt $P$. Client $i$ first performs local textual optimization,

$$P_i^{\mathrm{loc}}, \tau_i = \mathrm{TextGrad}\left(P, D_i^{\mathrm{tr}}\right), \tag{1}$$

where $\tau_i$ contains the textual feedback and accepted revisions produced during optimization.

**Common-prompt aggregation as a rate-constrained bottleneck.** Existing FedTextGrad methods (Chen et al., 2025) aggregate distributed textual updates into a single prompt shared by all clients. Concat retains more local content at a high token rate, whereas Summary and UID construct shorter common descriptions. Despite this difference, all impose one description on clients with potentially different objectives.

Rate–distortion provides a useful operational view of this bottleneck: prompt length acts as the rate, while client-specific task loss acts as semantic distortion (Nagle et al., 2024). For a prompt $Q$, define

$$R_\theta(Q) = c_\theta(Q), \qquad D_i(Q) = \mathbb{E}_{(x,y)\sim\mathcal{P}_i}\left[\ell(f_\theta(Q, x), y)\right], \tag{2}$$

where $c_\theta$ is the exact token count under the target model's tokenizer. Let $\mathcal{Q}_B = \{Q : R_\theta(Q) \leq B\}$ denote the feasible prompt class under token budget $B$. A common-prompt aggregator is character-

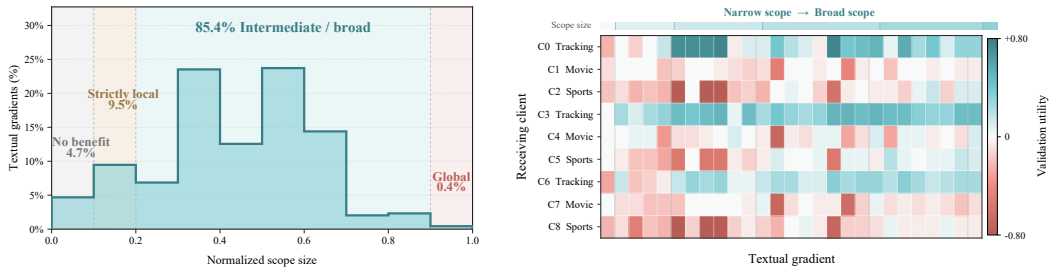



Figure 1: **Textual gradients have client-dependent scope. Left:** Distribution of empirical normalized scope across 2,070 rules under the default validation protocol. **Right:** Representative client–gradient utility matrix.

ized by

$$\mathcal{D}^{\star}_{\mathrm{com}}(B) = \min_{Q \in \mathcal{Q}_B} \sum_{i=1}^{K} w_i D_i(Q), \tag{3}$$

where $w_i \geq 0$ and $\sum_i w_i = 1$. Allowing client-conditioned descriptions instead gives

$$\mathcal{D}^{\star}_{\mathrm{cc}}(B) = \min_{Q_1,\ldots,Q_K \in \mathcal{Q}_B} \sum_{i=1}^{K} w_i D_i(Q_i). \tag{4}$$

Because the common-prompt class is recovered by imposing $Q_1 = \cdots = Q_K$,

$$\mathcal{D}^{\star}_{\mathrm{cc}}(B) \leq \mathcal{D}^{\star}_{\mathrm{com}}(B). \tag{5}$$

This is an expressivity relation rather than an algorithmic guarantee: client conditioning creates the freedom to retain different distributed information for different clients, but does not specify which updates should be delivered where.

**The missing client axis.** Rate-constrained aggregation determines which information is retained along the token axis, but federation introduces an orthogonal client axis: the same textual-gradient rule can induce different distortion changes on different clients. For a candidate rule $g$ and target client $i$, define its utility relative to the current shared prompt $P$ as

$$U_{ig} = D_i(P) - D_i(P \oplus g). \tag{6}$$

Thus, $U_{ig} > 0$ means that applying $g$ improves client $i$'s objective. For a pool of candidate rules $\mathcal{G} = \{g_1, \ldots, g_M\}$, stacking these utilities yields the client–gradient utility matrix

$$\mathbf{U} = [U_{ig}] \in \mathbb{R}^{K \times M}. \tag{7}$$

This matrix exposes what a single shared prompt leaves implicit: which textual updates are useful for which clients.

**Scope as distortion-reduction support.** For a utility threshold $\epsilon$, define

$$\mathbf{S} = \mathbf{1}\left[\mathbf{U} > \epsilon\right] \in \{0,1\}^{K \times M}. \tag{8}$$

The $g$-th column $\mathbf{s}_g = \mathbf{S}_{:g}$ defines the scope of textual gradient $g$: the set of clients for which the update reduces distortion by more than $\epsilon$. A textual gradient therefore has two complementary aspects: its content specifies *what should change*, while its scope specifies *where that change is useful*. Scope is not an intrinsic semantic property of a rule; it is defined relative to the current prompt, model, client objectives, and client distributions.

**Empirical evidence.** This client dimension is empirically substantial. Under our default operational definition, which uses $\widehat{U}_{ig} > 0$ with 20 validation examples per client, $4.7\%$ of rules benefit no client, $9.5\%$ are strictly local, $85.4\%$ benefit multiple but not all clients, and $0.4\%$ are classified as globally beneficial. In particular, requiring every client to cross a strict zero threshold makes the global category sensitive to finite validation noise and federation size. Appendix C.8 therefore complements these descriptive statistics with independent-resampling, confidence-adjusted, and uncertainty-aware analyses.

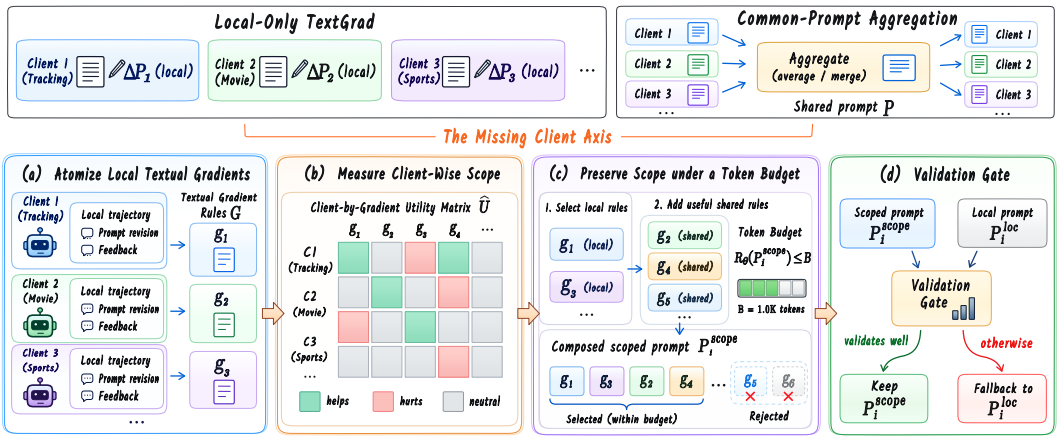



Figure 2: **Overview of Scoped-FedTextGrad.** Rather than collapsing them into a common prompt, we atomize local textual gradients, measure their client-wise scope, compose scope-compatible rules under a token budget, and optionally apply a validation gate against the local solution.

## 4 Federating Textual Gradients by Preserving Scope

We instantiate the preceding view in Scoped-FedTextGrad. Starting from a shared prompt $P$, clients first perform standard local TextGrad optimization. We then (i) atomize local textual updates into independently routable gradient units, (ii) estimate the client-wise utility and scope of each unit, and (iii) compose scope-compatible units under a prompt-rate constraint. An optional validation gate yields Safe-Scoped. Figure 2 summarizes the resulting framework.

### 4.1 Atomizing Local Textual Gradients

Local TextGrad revisions often bundle multiple reasoning prescriptions, making their transferability difficult to measure separately. We therefore extract a small set of independently routable textual-gradient units from each client's local optimization trajectory. Let $\tau_i$ denote the local TextGrad trajectory of client $i$, including its textual feedback and accepted prompt revisions. We define

$$\mathcal{G}_i = \mathcal{E}(\tau_i) = \{g_{i1}, \ldots, g_{im_i}\}, \qquad \mathcal{G} = \bigcup_{i=1}^{K} \mathcal{G}_i. \tag{9}$$

Each $g \in \mathcal{G}$ is a compact actionable instruction that can be evaluated and selected independently through the prompt interface. Importantly, atomization determines only *what* update is proposed; its applicability to target clients is determined entirely by measured utility in the next stage.

### 4.2 Measuring Where Each Gradient Applies

Section 3 defines the population utility of a textual gradient, but this quantity is not directly observable. We estimate it using each target client's local validation data. For client $i$,

$$\widehat{D}_i(Q) = \frac{1}{|D_i^{\mathrm{val}}|} \sum_{(x,y)\in D_i^{\mathrm{val}}} \ell(f_\theta(Q, x), y)\,. \tag{10}$$

The measured utility of rule $g$ for client $i$ is

$$\widehat{U}_{ig} = \widehat{D}_i(P) - \widehat{D}_i(P \oplus g). \tag{11}$$

These evaluations are performed on the target client's validation split; validation examples never leave the client. Stacking the measurements gives

$$\widehat{\mathbf{U}} = [\widehat{U}_{ig}] \in \mathbb{R}^{K\times M}, \tag{12}$$

with empirical scope

$$\widehat{\mathbf{S}} = \mathbf{1}[\widehat{\mathbf{U}} > \epsilon]. \tag{13}$$

The candidate set eligible for client $i$ is therefore

$$\mathcal{H}_i = \{g \in \mathcal{G} : \widehat{U}_{ig} > \epsilon\}. \tag{14}$$

Eligibility depends on measured target-client benefit rather than source task identity, textual similarity, or global frequency. In our setting, distortion is instantiated by 0–1 loss, making Eq. (11) equivalent to validation-accuracy improvement. The utility matrix is therefore an update-centric representation of personalization: columns describe the transferability of reusable textual gradients rather than merely assigning each client an independent final prompt.

### 4.3 Preserving Scope under a Token Budget

Scope determines which rules are eligible for a client, while the prompt-rate constraint determines how much eligible information can be retained. A natural first-order rate-constrained objective is

$$\begin{aligned} \max_{\mathcal{R}_i \subseteq \mathcal{G}} \quad & \sum_{g \in \mathcal{R}_i} \widehat{U}_{ig} \\ \text{s.t.} \quad & \widehat{U}_{ig} > \epsilon, \qquad \forall g \in \mathcal{R}_i, \\ & R_\theta\left(P \oplus \mathrm{Compose}(\mathcal{R}_i)\right) \le B. \end{aligned} \tag{15}$$

This objective uses singleton rule utilities as a practical first-order criterion and does not assume that utilities remain additive after composition. Rather than solving the discrete selection problem exactly, we use a deterministic scope-aware composition policy. Among eligible rules, those originating from the target client are considered first to preserve locally grounded improvements. Remaining rules are ranked by measured utility per token,

$$\rho_i(g) = \frac{\widehat{U}_{ig}}{c_\theta(g)}, \tag{16}$$

and retained while the complete composed prompt remains within the token budget. The resulting prompt is

$$P_i^{\mathrm{scope}} = P \oplus \mathrm{Compose}(\mathcal{R}_i), \qquad R_\theta(P_i^{\mathrm{scope}}) \le B. \tag{17}$$

### 4.4 Validation Gating under Scope Uncertainty

Scope estimates are obtained from finite validation data, and individually beneficial rules can interact after composition. We therefore optionally compare the complete scoped prompt against the independently optimized local solution:

$$P_i^{\mathrm{final}} = \begin{cases} P_i^{\mathrm{loc}}, & \widehat{D}_i(P_i^{\mathrm{scope}}) > \widehat{D}_i(P_i^{\mathrm{loc}}) + \delta, \\ P_i^{\mathrm{scope}}, & \text{otherwise}. \end{cases} \tag{18}$$

We refer to the ungated and gated variants as Scoped-FedTextGrad and Safe-Scoped, respectively. The gate uses validation data only and never consults test labels.

The resulting framework separates three decisions: scope estimation determines *where* a textual gradient is useful, rate-constrained composition determines *how much* useful information is retained, and validation gating determines *whether* the composed description should replace the local solution.

## 5 Experiments

Section 3 establishes that textual gradients predominantly exhibit partial client scope. We next evaluate whether preserving this structure improves federated textual optimization and whether the resulting advantage is robust to prompt rate, design choices, and model/task variation. Specifically, we study four research questions:

- **RQ1:** Does preserving textual-gradient scope improve federated textual optimization?
- **RQ2:** Does the advantage persist under matched prompt rates?
- **RQ3:** What drives the gain, and how does validation gating affect negative transfer?
- **RQ4:** Does the framework generalize across model families, model scales, and tasks?

Table 1: **Main Qwen3-8B results.** Best results are highlighted in bold and second-best results are underlined. Local TextGrad serves as the reference for NTR and is excluded from NTR ranking. All improvements over the baselines are statistically significant across five random seeds under a two-sided $t$-test ($p \leq 0.05$).

| | Task Skew | | | | Compound Skew | | | |
|---|---|---|---|---|---|---|---|---|
| Method | AvgAcc ↑ | Worst ↑ | NTR ↓ | Tokens ↓ | AvgAcc ↑ | Worst ↑ | NTR ↓ | Tokens ↓ |
| Original | $0.596_{\pm 0.007}$ | $\underline{0.459}_{\pm 0.017}$ | $\underline{0.285}_{\pm 0.055}$ | 42.0 | $0.595_{\pm 0.006}$ | $0.417_{\pm 0.018}$ | $\underline{0.308}_{\pm 0.060}$ | 42.0 |
| Local TextGrad | $0.640_{\pm 0.024}$ | $0.397_{\pm 0.045}$ | $0.000_{\pm 0.000}$ | 89.3 | $0.658_{\pm 0.027}$ | $0.428_{\pm 0.034}$ | $0.000_{\pm 0.000}$ | 88.8 |
| FedTextGrad-Concat | $0.632_{\pm 0.046}$ | $0.319_{\pm 0.095}$ | $0.389_{\pm 0.083}$ | 667.5 | $0.583_{\pm 0.082}$ | $0.321_{\pm 0.092}$ | $0.498_{\pm 0.115}$ | 665.2 |
| FedTextGrad-Summary | $0.516_{\pm 0.031}$ | $0.188_{\pm 0.071}$ | $0.604_{\pm 0.078}$ | 192.3 | $0.582_{\pm 0.026}$ | $0.346_{\pm 0.049}$ | $0.572_{\pm 0.058}$ | 204.0 |
| FedTextGrad-UID | $0.514_{\pm 0.054}$ | $0.198_{\pm 0.079}$ | $0.620_{\pm 0.103}$ | 205.3 | $0.502_{\pm 0.049}$ | $0.243_{\pm 0.057}$ | $0.646_{\pm 0.068}$ | 200.5 |
| SCOPED-FEDTEXTGRAD | $\underline{0.699}_{\pm 0.026}$ | $0.455_{\pm 0.045}$ | $0.303_{\pm 0.064}$ | 183.7 | $\underline{0.730}_{\pm 0.023}$ | $\underline{0.493}_{\pm 0.036}$ | $\underline{0.308}_{\pm 0.068}$ | 175.5 |
| SAFE-SCOPED | $\mathbf{0.729}_{\pm 0.023}$ | $\mathbf{0.526}_{\pm 0.024}$ | $\mathbf{0.134}_{\pm 0.062}$ | 163.4 | $\mathbf{0.747}_{\pm 0.019}$ | $\mathbf{0.512}_{\pm 0.031}$ | $\mathbf{0.127}_{\pm 0.042}$ | 155.8 |

### 5.1 EXPERIMENTAL SETUP

**Tasks & Datasets.** Our main experiments use three BIG-Bench Hard (BBH) datasets: **tracking shuffled objects**, **movie recommendation**, and **sports understanding**. We additionally evaluate on **reasoning about colored objects**, **salient translation error detection**, **temporal sequences**, and **web of lies**. We consider two heterogeneous federations. In *Task Skew*, each client contains examples from exactly one task, with tasks rotated across clients. In *Compound Skew*, we retain the same task assignment and additionally introduces within-task length-based complexity skew. All clients receive equal numbers of examples, and no examples are shared between clients. Additional setup details are provided in the Appendix A.

**Base LLMs & Baseline Methods.** Qwen3-8B is the primary model. To test scaling and cross-family transfer, we additionally evaluate Qwen3-4B, Qwen3-14B, Qwen3-32B, Llama-3.1-8B, and Phi-3.5-mini under a common reduced protocol. All models use deterministic decoding and their exact native tokenizer. We compare against **Local TextGrad**, which retains each client's locally optimized prompt, and the FedTextGrad global aggregators **Concat**, **Summary**, and **UID** (Chen et al., 2025). For rate-matched comparisons, we additionally use **Truncated-Concat** and **LLM-Compressed-Concat**. For ablations, we remove measured utility (**w/o Utility**), replace it with lexical similarity (**Text-Similarity**), or restrict routing to locally sourced rules (**Local-only**).

**Evaluation Metrics.** We report mean client test accuracy (**AvgAcc**), worst-client accuracy (**Worst**), mean final prompt length (**Tokens**), and negative-transfer rate (**NTR**), defined as the fraction of clients whose test accuracy is lower than Local TextGrad.

**Implementation Details.** In the main implementation, each client contributes at most three candidate rules, and each rule is capped at 64 model-tokenizer tokens. We set $\epsilon = 0$, such that only rules with strictly positive measured utility are eligible. The default prompt budget is $B = 256$ tokens. The validation gate uses $\delta = 0$. The complete rule-extraction prompt, deterministic post-processing procedure, rule-card schema, high-level pseudocode, communication analysis, and multi-round extension are provided in Appendix A.3.

### 5.2 DOES PRESERVING SCOPE IMPROVE FEDERATED TEXTUAL OPTIMIZATION? (RQ1)

Table 1 presents the primary results. Across both heterogeneity settings, SCOPED-FEDTEXTGRAD consistently outperforms Local TextGrad and all FedTextGrad aggregation baselines, with gains of up to 22.8 percentage points in AvgAcc. It also achieves these gains with substantially shorter prompts, reducing prompt tokens by roughly 73% relative to unconstrained concatenation at the default budget. SAFE-SCOPED further improves average and worst-client performance while reducing NTR. These results demonstrate that preserving client-dependent gradient scope substantially improves both the effectiveness and reliability of federated textual optimization. As shown in Appendix C.2, SCOPED-FEDTEXTGRAD consistently outperforms baselines across different federated communication rounds.

### 5.3 DOES THE ADVANTAGE PERSIST UNDER MATCHED PROMPT RATES? (RQ2)

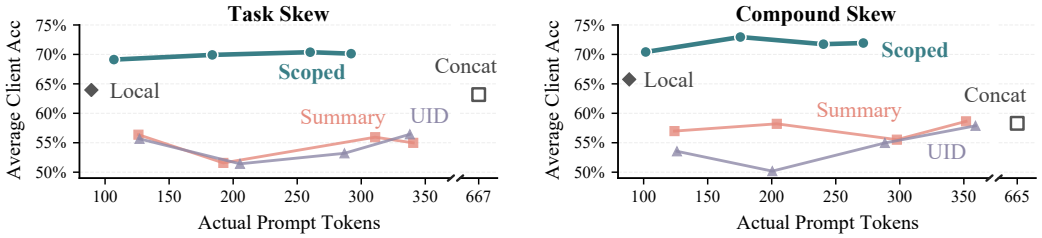



Figure 3: **Scope preservation improves the accuracy–rate frontier.** Local TextGrad and unconstrained Concat are shown as reference operating points.

Our rate–distortion view suggests that aggregation methods should be compared by the accuracy they achieve at a given prompt rate. Figure 3 already shows that SCOPED-FEDTEXTGRAD dominates Summary and UID across the full tested rate range. For SCOPED-FEDTEXTGRAD, Summary, and UID, each connected point corresponds to a budget in $\{128, 256, 512, 1024\}$; Local TextGrad and unconstrained Concat provide low- and high-rate reference operating points. Across both heterogeneity settings, SCOPED-FEDTEXTGRAD consistently achieves a stronger accuracy–rate trade-off than Summary and UID, maintaining high accuracy over a broad range of prompt rates.

Table 2: **Equal-rate controls under a 256-token ceiling.** We report AvgAcc for all methods.

| Method | Task Skew | Compound Skew |
|---|---|---|
| SCOPED-FEDTEXTGRAD | **0.699** | **0.730** |
| Truncated-Concat | 0.687 | 0.590 |
| LLM-Compressed-Concat | 0.644 | 0.539 |

Table 2 further controls for rate by comparing against two 256-token alternatives that retain or compress the same aggregated information. Under Task Skew, Truncated-Concat is competitive with SCOPED-FEDTEXTGRAD, whereas under Compound Skew the gap widens substantially. Thus, the gains of scope-preserving composition persist under matched rate and cannot be explained by a larger prompt budget.

### 5.4 WHAT DRIVES THE GAIN AND REDUCES NEGATIVE TRANSFER? (RQ3)

Table 3 shows that each component contributes meaningfully to the final performance. Directly measuring target-client utility is critical for identifying gradient scope, while lexical similarity is a poor substitute. Cross-client transfer also remains important: restricting each client to its own locally originated rules consistently weakens performance, showing that the method benefits from selectively reusing useful updates across clients. Finally, validation gating substantially improves robustness by suppressing residual negative transfer introduced by noisy utility estimates and multi-rule composition. Together, these results support the full design of scope-aware routing with measured utility, cross-client sharing, and validation gating. Additional controls in Appendix C further separate scope preservation from generic personalization. Global utility ranking and oracle task clustering remain weaker than client-specific scope preservation, while a query-matched personalized greedy search receives the same validation-query budget but does not recover the accuracy gains of SCOPED-FEDTEXTGRAD.

Table 3: **Ablation results.**

| | Task Skew | | Compound Skew | |
|---|---|---|---|---|
| Variant | Acc. ↑ | NTR ↓ | Acc. ↑ | NTR ↓ |
| SCOPED-FEDTEXTGRAD | 0.699 | 0.303 | 0.730 | 0.308 |
| w/o Utility | 0.484 | 0.650 | 0.541 | 0.569 |
| Text-Similarity | 0.447 | 0.669 | 0.508 | 0.558 |
| Local-only | 0.646 | 0.280 | 0.682 | 0.238 |
| SAFE-SCOPED | **0.729** | **0.134** | **0.747** | **0.127** |

### 5.5 DOES THE FRAMEWORK GENERALIZE ACROSS MODELS, SCALES, AND TASKS? (RQ4)

**Generalization across model scales and families.** Figure 4 shows that the gains extend well beyond Qwen3-8B. Across Qwen3 models from 4B to 32B and two additional model families, SAFE-SCOPED achieves the highest observed AvgAcc under both heterogeneity settings. While UID becomes increasingly competitive at larger Qwen scales, validation-gated scope preservation remains consistently effective across model capacities and families.

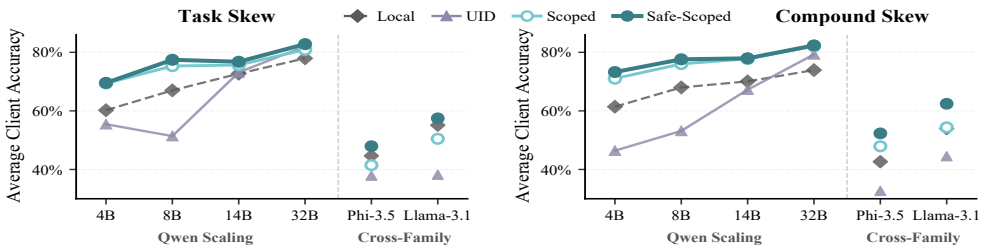



Figure 4: **Generalization across model scales and families.** Average client accuracy under Task Skew (left) and Compound Skew (right). Connected points show Qwen3 scaling from 4B to 32B, while isolated points report cross-family results on Phi-3.5-mini and Llama-3.1-8B.

**Generalization to additional tasks.** We further evaluate four additional BBH tasks under Compound Skew. As shown in Figure 5, SCOPED-FEDTEXTGRAD and SAFE-SCOPED consistently deliver substantially stronger cross-client transfer than the global aggregation baselines. Across the four-task suite, they improve macro AvgAcc from 0.657 for Local TextGrad to 0.819 and 0.815, respectively, compared with 0.610 for Summary and 0.588 for UID. Together, these results demonstrate that scope-preserving federation generalizes across both model and task shifts.

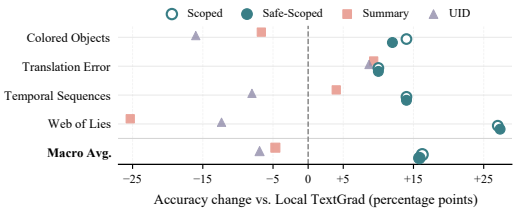



Figure 5: **Generalization to additional tasks.** Accuracy changes are measured relative to Local TextGrad; positive values indicate improvement over local optimization.

**Same-task non-IID transfer.** We further remove task-identity heterogeneity by constructing federations in which all clients solve the same BBH task and differ only in within-task input-complexity distribution. Both SCOPED-FEDTEXTGRAD and SAFE-SCOPED continue to improve macro AvgAcc, showing that scope-aware sharing does not rely on cross-task heterogeneity. Full results are reported in Appendix C.11.

**More Results.** Appendix C further reports hyperparameter sensitivity, client-count scaling, scope analyses, query-matched controls, and complete model- and task-level results. A sparse variant that probes only eight cross-client candidates preserves nearly all of the gain while roughly halving validation inference, suggesting that exhaustive $K \times M$ probing is not necessary.

## 6 CONCLUSION

We studied federated textual optimization through the client-dependent *scope* of textual gradients. Our empirical analysis shows that textual updates are typically transferable across clients while exhibiting substantial client-dependent variation in utility, revealing a client axis that common-prompt aggregation leaves implicit. Building on this observation, we introduced SCOPED-FEDTEXTGRAD, which atomizes local updates, measures their target-client utility, and preserves their scope during rate-constrained prompt composition. The validation-gated SAFE-SCOPED further guards against harmful compositions by reverting to the independently optimized local prompt when necessary. Across heterogeneous federations, client scales, token budgets, tasks, and LLM families, scope-preserving aggregation consistently improves the accuracy–rate trade-off and substantially reduces negative transfer. More broadly, our results suggest that textual gradients should not be viewed only as descriptions of *what* to change: in distributed optimization, their usefulness also has a measurable *where*. Making this client-dependent applicability explicit provides a simple principle for designing more selective and efficient federated textual optimization systems.

### AI USE STATEMENT

Generative AI tools were used only for spelling and grammar checks and for limited language polishing. They were not used to design the methodology, conduct experiments, analyze results, or determine scientific claims. All AI-assisted edits were reviewed by the authors, who take full responsibility for the final content of the paper.

### ETHICS STATEMENT

This work uses publicly available benchmark datasets and publicly released language models, and does not involve human-subject studies or the collection of personal or sensitive data. We are not aware of additional ethical concerns specific to this work.

### REPRODUCIBILITY STATEMENT

We provide detailed dataset construction, experimental protocols, model and inference settings, baseline implementations, algorithmic details, and additional analyses in the main paper and appendix. The code for reproducing our experiments is available in the anonymous repository at https://anonymous.4open.science/r/Scope-FedTextual.

## REFERENCES

Lakshya A. Agrawal, Shangyin Tan, Dilara Soylu, Noah Ziems, Rishi Khare, Krista Opsahl-Ong, Arnav Singhvi, Herumb Shandilya, Michael J. Ryan, Meng Jiang, Christopher Potts, Koushik Sen, Alex Dimakis, Ion Stoica, Dan Klein, Matei Zaharia, and Omar Khattab. GEPA: Reflective prompt evolution can outperform reinforcement learning. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://proceedings.iclr.cc/paper_files/paper/2026/hash/0e9e708b6f48e14fd0ac29e167413f76-Abstract-Conference.html.

Weihao Bo, Yanpeng Sun, Yu Wang, Xinyu Zhang, and Zechao Li. FedMGP: Personalized federated learning with multi-group text-visual prompts. In *Advances in Neural Information Processing Systems*, volume 38, 2025. doi: 10.52202/085713-1726. URL https://proceedings.neurips.cc/paper_files/paper/2025/hash/4a4d5fca4bbb93ead4548854f1f28c0f-Abstract-Conference.html.

Minghui Chen, Ruinan Jin, Wenlong Deng, Yuanyuan Chen, Zhi Huang, Han Yu, and Xiaoxiao Li. Can textual gradient work in federated learning? In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/hash/1a25185fe1b0d46241de7738ea320710-Abstract-Conference.html.

Minghui Chen, Wenlong Deng, James Y. Zou, Han Yu, and Xiaoxiao Li. Textual equilibrium propagation for deep compound AI systems. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://proceedings.iclr.cc/paper_files/paper/2026/hash/f5a8b5e5d007e66c929b971c2bc21d76-Abstract-Conference.html.

Chengying Fang, Wenke Huang, Guancheng Wan, Yihao Yang, and Mang Ye. FedPHA: Federated prompt learning for heterogeneous client adaptation. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 15960–15975. PMLR, 2025. URL https://proceedings.mlr.press/v267/fang25e.html.

Chrisantha Fernando, Dylan Sunil Banarse, Henryk Michalewski, Simon Osindero, and Tim Rocktäschel. Promptbreeder: Self-referential self-improvement via prompt evolution. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 13481–13544. PMLR, 2024. URL https://proceedings.mlr.press/v235/fernando24a.html.

Tao Guo, Song Guo, Junxiao Wang, Xueyang Tang, and Wenchao Xu. PromptFL: Let federated participants cooperatively learn prompts instead of models—federated learning in age of foundation model. *IEEE Transactions on Mobile Computing*, 23(5):5179–5194, 2024. doi: 10.1109/TMC.2023.3302410. URL https://doi.org/10.1109/TMC.2023.3302410.

Omar Khattab, Arnav Singhvi, Paridhi Maheshwari, Zhiyuan Zhang, Keshav Santhanam, Sri Vardhamanan A, Saiful Haq, Ashutosh Sharma, Thomas Joshi, Hanna Moazam, Heather Miller, Matei Zaharia, and Christopher Potts. DSPy: Compiling declarative language model calls into state-of-the-art pipelines. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://proceedings.iclr.cc/paper_files/paper/2024/hash/f1cf02ce09757f57c3b93c0db83181e0-Abstract-Conference.html.

Hongxia Li, Wei Huang, Jingya Wang, and Ye Shi. Global and local prompts cooperation via optimal transport for federated learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 12151–12161, 2024. URL https://openaccess.thecvf.com/content/CVPR2024/html/Li_Global_and_Local_Prompts_Cooperation_via_Optimal_Transport_for_Federated_CVPR_2024_paper.html.

Tian Li, Anit Kumar Sahu, Manzil Zaheer, Maziar Sanjabi, Ameet Talwalkar, and Virginia Smith. Federated optimization in heterogeneous networks. In *Proceedings of Machine Learning and Systems*, volume 2, pp. 429–450, 2020. URL https://proceedings.mlsys.org/paper/2020/hash/1f5fe83998a09396ebe6477d9475ba0c-Abstract.html.

Tian Li, Shengyuan Hu, Ahmad Beirami, and Virginia Smith. Ditto: Fair and robust federated learning through personalization. In *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pp. 6357–6368. PMLR, 2021. URL https://proceedings.mlr.press/v139/li21h.html.

Jun Luo, Chen Chen, and Shandong Wu. Mixture of experts made personalized: Federated prompt learning for vision-language models. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/hash/92369a01fbe8046a093746389b2c413e-Abstract-Conference.html.

Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Agüera y Arcas. Communication-efficient learning of deep networks from decentralized data. In *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics*, volume 54 of *Proceedings of Machine Learning Research*, pp. 1273–1282. PMLR, 2017. URL https://proceedings.mlr.press/v54/mcmahan17a.html.

Alliot Nagle, Adway Girish, Marco Bondaschi, Michael Gastpar, Ashok Vardhan Makkuva, and Hyeji Kim. Fundamental limits of prompt compression: A rate-distortion framework for black-box language models. In *Advances in Neural Information Processing Systems*, volume 37, 2024. doi: 10.52202/079017-3009. URL https://proceedings.neurips.cc/paper_files/paper/2024/hash/ac8fbba029dadca99d6b8c3f913d3ed6-Abstract-Conference.html.

Reid Pryzant, Dan Iter, Jerry Li, Yin Lee, Chenguang Zhu, and Michael Zeng. Automatic prompt optimization with "gradient descent" and beam search. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pp. 7957–7968. Association for Computational Linguistics, 2023. doi: 10.18653/v1/2023.emnlp-main.494. URL https://aclanthology.org/2023.emnlp-main.494/.

Zhuang Qi, Yu Pan, Lei Meng, Sijin Zhou, Han Yu, Xiaoxiao Li, and Xiangxu Meng. Global prompt refinement with non-interfering attention masking for one-shot federated learning. In *Advances in Neural Information Processing Systems*, volume 38, 2025. doi: 10.52202/085713-0405. URL https://proceedings.neurips.cc/paper_files/paper/2025/hash/11e7d0a3341de7f451141248dfbff4d3-Abstract-Conference.html.

Yao Shu, Wenyang Hu, See-Kiong Ng, Bryan Kian Hsiang Low, and Fei Yu. Ferret: Federated full-parameter tuning at scale for large language models. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 55379–55402. PMLR, 2025. URL https://proceedings.mlr.press/v267/shu25a.html.

Ganyu Wang, Jinjie Fang, Maxwell Juncheng Yin, Bin Gu, Xi Chen, Boyu Wang, Yi Chang, and Charles Ling. FedOne: Query-efficient federated learning for black-box discrete prompt learning. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 62857–62889. PMLR, 2025. URL https://proceedings.mlr.press/v267/wang25ab.html.

Hanyang Wang, Lu Wang, Chaoyun Zhang, Tianjun Mao, Si Qin, Qingwei Lin, Saravan Rajmohan, and Dongmei Zhang. Text2Grad: Reinforcement learning from natural language feedback. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://proceedings.iclr.cc/paper_files/paper/2026/hash/52fbb930ec6941099f1384321d8b9461-Abstract-Conference.html.

Yunlu Yan, Chun-Mei Feng, Wangmeng Zuo, Rick Siow Mong Goh, Yong Liu, and Lei Zhu. Federated residual low-rank adaptation of large language models. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/hash/906c860f1b7515a8ffec02dcdac74048-Abstract-Conference.html.

Chengrun Yang, Xuezhi Wang, Yifeng Lu, Hanxiao Liu, Quoc V. Le, Denny Zhou, and Xinyun Chen. Large language models as optimizers. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://proceedings.iclr.cc/paper_files/paper/2024/hash/3339f19c5fcee3ad74502947a32be9e6-Abstract-Conference.html.

Fu-En Yang, Chien-Yi Wang, and Yu-Chiang Frank Wang. Efficient model personalization in federated learning via client-specific prompt generation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 19159–19168, 2023. URL https://openaccess.thecvf.com/content/ICCV2023/html/Yang_Efficient_Model_Personalization_in_Federated_Learning_via_Client-Specific_Prompt_Generation_ICCV_2023_paper.html.

Mert Yuksekgonul, Federico Bianchi, Joseph Boen, Sheng Liu, Pan Lu, Zhi Huang, Carlos Guestrin, and James Zou. Optimizing generative AI by backpropagating language model feedback. *Nature*, 639:609–616, 2025. doi: 10.1038/s41586-025-08661-4. URL https://doi.org/10.1038/s41586-025-08661-4.

Peiyan Zhang, Haibo Jin, Leyang Hu, Xinnuo Li, Liying Kang, Man Luo, Yangqiu Song, and Haohan Wang. Revolve: Optimizing AI systems by tracking response evolution in textual optimization. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 75216–75233. PMLR, 2025. URL https://proceedings.mlr.press/v267/zhang25aj.html.

Haodong Zhao, Wei Du, Fangqi Li, Peixuan Li, and Gongshen Liu. FedPrompt: Communication-efficient and privacy-preserving prompt tuning in federated learning. In *ICASSP 2023 – 2023 IEEE International Conference on Acoustics, Speech and Signal Processing*, pp. 1–5. IEEE, 2023. doi: 10.1109/ICASSP49357.2023.10095356. URL https://doi.org/10.1109/ICASSP49357.2023.10095356.

# CONTENTS

**A. Detailed Experimental Setup** ..... **14**
A.1. Datasets and Client Construction ..... 14
A.2. Models and Inference ..... 15
A.3. Scoped-FedTextGrad Implementation ..... 16
A.4. Evaluation Metrics ..... 20
**B. Finite-Sample Concentration Analysis** ..... **20**
**C. Additional Experimental Results** ..... **21**
C.1. Generalization with the Number of Clients ..... 21
C.2. Generalization with Different Federated Communication Rounds ..... 22
C.3. Hyperparameter Sensitivity ..... 23
C.4. Validation-Gate Sensitivity ..... 24
C.5. Full Additional-Task Results ..... 24
C.6. Full Model-Scale and Cross-Family Results ..... 24
C.7. Scope across Models and Client Counts ..... 25
C.8. Finite-Validation Stability of Gradient Scope ..... 25
C.9. Additional Selective-Sharing Controls ..... 26
C.10. Sparse Scope Estimation ..... 27
C.11. Same-Task Non-IID Federations ..... 27
C.12. Rationale for the Length-Based Difficulty Proxy ..... 28
C.13. Representative Routing Cases ..... 28
**D. Limitations and Future Work** ..... **29**
**E. Licenses for Existing Assets** ..... **29**

# A Detailed Experimental Setup

## A.1 Datasets and Client Construction

**Main tasks.** Our primary experiments use three BIG-Bench Hard (BBH) tasks: **tracking shuffled objects**, **movie recommendation**, and **sports understanding**. For each task, the examples in the official BBH file are partitioned by their original zero-based indices: indices 0–49 form the training source pool, 50–149 form the validation source pool, and 150 onward form the test source pool. Each task–split pool is shuffled independently and deterministically using the run seed before client allocation. Allocation is without replacement, so training, validation, and test examples are disjoint, and no example is shared by two clients within a run.

For $K = 6$, two clients are assigned to each task and each client receives 20 training, 20 validation, and 50 test examples. For $K = 9$, three clients are assigned to each task and each client receives 16 training, 20 validation, and 33 test examples. The reduced training size for $K = 9$ follows from the 50-example per-task training pool and guarantees disjoint client data.

**Task Skew.** Let $T_0, T_1, T_2$ denote the three main tasks. Under Task Skew, client $i$ is assigned

$$\text{task}(i) = T_{i \bmod 3}. \tag{19}$$

Thus, heterogeneity arises solely from task identity. Clients assigned to the same task receive disjoint examples without any difficulty-based ordering.

**Compound Skew.** Under Compound Skew, we retain the same task assignment but additionally introduce within-task complexity skew. Since BBH does not provide unified instance-level complexity annotations, we use normalized question length as a fixed, label-free proxy for the amount of contextual and task-relevant information presented to the model,

$$c(x) = \min\left(1, \frac{\text{question word count}}{100}\right). \tag{20}$$

Clients are assigned to short- or long-input strata according to

$$\text{stratum}(i) = \begin{cases} \text{short}, & \lfloor i/3 \rfloor \bmod 2 = 0, \\ \text{long}, & \lfloor i/3 \rfloor \bmod 2 = 1. \end{cases} \tag{21}$$

Within each task and split, short-stratum clients receive the shortest remaining examples, whereas long-stratum clients receive the longest remaining examples. This induces a deterministic within-task distribution shift without relying on model errors or label-derived difficulty estimates. Example identifiers are tracked during allocation to prevent overlap.

**Additional tasks.** The main-paper task-generalization experiment evaluates Qwen3-8B on four additional BBH tasks: **reasoning about colored objects**, **salient translation error detection**, **temporal sequences**, and **web of lies**. These experiments use Compound Skew, $K = 6$, three seeds, three local TextGrad steps, local batch size 3, and a 256-token prompt budget.

**Client-count scaling.** To examine scaling with federation size, we additionally vary

$$K \in \{5, 10, 15, 20, 25, 30\}.$$

These experiments fix Qwen3-8B, three local TextGrad steps, local batch size 3, and a 256-token prompt budget. To maintain sufficient disjoint data at larger $K$, every client receives 5 training, 10 validation, and 10 test examples. The allocation procedure guarantees no within-run client overlap, while task counts differ by at most one client.

### A.2 Models and Inference

The model-scaling and cross-family experiments use a common reduced protocol: two heterogeneity settings, $K \in \{6, 9\}$, five seeds, three local TextGrad steps, local batch size 3, and a 256-token prompt budget. Each model–setting combination therefore averages six runs.

All models use deterministic decoding with temperature 0, `top_p` $= 1$, and `top_k` $= -1$. Qwen thinking mode is disabled. Prompt length is always measured using the exact native tokenizer of the corresponding model. Model requests are served through an OpenAI-compatible local inference interface and cached deterministically.

We use a strict answer parser requiring an explicit `Answer:` field. The parser accepts answer letters, parenthesized choices, dollar signs, and exact option text.

#### A.2.1 Baseline and Control Implementations

**Original.** Original denotes the unoptimized round-start prompt $P$.

**Truncated-Concat.** We first construct the standard FedTextGrad concatenated prompt and then apply the target model's exact tokenizer, retaining the longest prefix whose total prompt length does not exceed $B$ tokens.

**LLM-Compressed-Concat.** We provide the same concatenated textual information to the compression model and request a deterministic summary whose final serialized prompt does not exceed $B$ tokens. The compressed prompt is post-checked with the exact target-model tokenizer, and no validation or test labels are provided to the compressor.

**w/o Utility.** This ablation removes target-client utility from routing. All extracted rules are treated as eligible, while the same local-first composition policy and token budget are retained. It tests whether atomization and recomposition alone can explain the gain.

**Text-Similarity.** This ablation replaces measured target-client utility with textual similarity between a candidate rule and the target client's local textual-gradient trajectory. The similarity score is used for rule filtering and ranking, while the remaining routing and budget constraints are unchanged. It tests whether simple textual relevance can substitute for measured utility.

**Local-only.** This ablation disables cross-client transfer by restricting each client to rules originating from itself. Measured-utility filtering and the same token-budgeted composition procedure are otherwise retained. It isolates the contribution of selectively reusing useful rules from other clients.

**Global-Utility.** For each candidate rule $g$, we average its measured utility over clients,

$$\bar{U}_g = \frac{1}{K} \sum_{i=1}^{K} \widehat{U}_i(g),$$

and construct a single common prompt by greedily selecting positive-utility rules according to $\bar{U}_g / c_\theta(g)$ under the same 256-token budget. The resulting prompt is broadcast unchanged to every client.

**Task-Cluster.** This oracle control groups clients by known BBH task identity and performs the same utility-based rule composition independently within each task group. It therefore tests whether the advantage of scope preservation can be reduced to task-level clustering.

**Val-Select(Local, UID).** For each client, we evaluate its independently optimized Local TextGrad prompt and the UID global prompt on local validation data and deploy whichever has higher validation accuracy. No test labels are used.

**Query-Matched Personalized Greedy.** This baseline performs generic client-specific validation-based routing without explicitly estimating rule scope. For each client, candidate rules are processed in a fixed order and each rule is evaluated once when appended to the current personalized prompt. The rule is retained whenever it improves the client's validation objective and fits within the token budget. It receives the same candidate pool and approximately the same target-client validation-query budget as SCOPED-FEDTEXTGRAD.

**Sparse-Scoped.** Sparse-Scoped reduces exhaustive client–rule probing through a cheap candidate pre-filter. For each target client, all locally sourced rules are retained, while cross-client rules are ranked by the same textual-similarity score used in the Text-Similarity ablation. Only the top-$L$ cross-client candidates are subsequently evaluated using the true target-client validation utility. Utility filtering, utility-per-token ranking, token-budgeted composition, and validation gating are otherwise unchanged. Unlike Text-Similarity, similarity is used only for candidate screening and never determines the final routing decision.

For each evaluated backbone, rule extraction and LLM-based compression use the same target language model as the downstream prompt optimization.

## A.3 SCOPED-FEDTEXTGRAD IMPLEMENTATION

### A.3.1 RULE EXTRACTION AND RULE-CARD CONSTRUCTION

A rule card is constructed from each accepted local TextGrad revision. The extraction model does *not* generate the complete serialized rule card; it converts the textual-gradient feedback associated with the accepted revision and the corresponding prompt change into a compact textual rule.

For each accepted local revision, we retain two textual inputs. The first is the complete textual-gradient feedback associated with the revision, denoted *failure mode* in our implementation. The second is a preliminary rule obtained by comparing the accepted candidate prompt against the prompt before the revision and retaining only newly introduced content.

Specifically, repeated lines and redundant answer-format instructions are removed, list prefixes are stripped, and the remaining additions are merged. If the candidate prompt introduces no identifiable new content, we fall back to the textual feedback itself.

The resulting representation therefore separates the *textual update* from its experimentally measured *scope*: source-side metadata describes where the rule originated, whereas target-client utility determines where the rule is actually useful.

### A.3.2 RULE-EXTRACTION PROMPT AND POST-PROCESSING

The extraction model receives the original textual-gradient feedback together with the preliminary update and is instructed to produce exactly one compact, portable reasoning rule.

**Prompt for Rule Extraction**

```
Convert the local textual-gradient feedback below into exactly
one portable reasoning rule.  Return only one imperative sentence,
with no analysis, labels, bullets, examples, task names, or answer.
The rule must be at most {max_rule_tokens} tokens and preserve the
concrete fix.

Failure mode:
{failure_mode}

Local update candidate:
{proposed_rule}
```

The three placeholders denote the complete local textual-gradient feedback, the preliminary rule extracted from the candidate prompt update, and the maximum rule length, respectively. We use a maximum of 64 model-tokenizer tokens in the main experiments.

Table 4: **Serialized rule-card schema.** Only `proposed_rule` is generated by the extraction model; all remaining fields are deterministically populated from the local optimization trajectory or metadata.

| Field | Type | Description |
|---|---|---|
| `card_id` | string | Unique identifier of the form {`source_client`}_g{`step`}. |
| `source_client` | string | Client that produced the local textual gradient. |
| `failure_mode` | string | Complete textual-gradient feedback before compact extraction. |
| `proposed_rule` | string | Final normalized and token-capped rule used for utility evaluation. |
| `applicable_condition` | string | Source-side provenance description; not used by the main router. |
| `evidence_examples_ids` | list[string] | Local batch examples exposed to the feedback engine. |
| `local_score_before` | number | Local batch accuracy before the accepted revision. |
| `local_score_after` | number | Local batch accuracy after the accepted revision. |
| `task` | string | Source task retained for compatibility with earlier artifacts. |
| `local_delta` | number | Local batch accuracy improvement of the accepted revision. |
| `token_length` | integer | Exact tokenizer length of the final rule. |
| `source_task` | string | Explicit source-task field, identical to `task`. |

The prompt asks the model to return a single imperative rule without analysis, task names, examples, answer labels, or task-specific answers. These are linguistic instructions rather than separately verified syntactic constraints; in particular, we do not run an additional classifier to enforce that the output contains exactly one grammatical sentence.

**Deterministic post-processing.** The returned text is normalized deterministically before being used for routing. We first remove surrounding whitespace, explicit `<think>` tags, and leading labels such as `Rule:` or `Answer:`. Nonempty lines are then flattened into a single string after removing common list markers. If the normalized extraction is empty, we fall back to the preliminary rule.

Finally, the rule is subjected to a hard token cap using the exact tokenizer of the evaluated model. If the output exceeds the configured limit, we retain the longest prefix that fits the budget, preferentially ending at a word boundary, remove dangling punctuation, and add terminal punctuation when needed. The exact tokenizer is then applied again to record the final rule length.

### A.3.3 Serialized Rule-Card Schema

The extraction model produces only the `proposed_rule` field. The remaining fields are generated deterministically and stored together with the rule for provenance and analysis. Table 4 gives the complete serialized schema used in our experiments.

The `applicable_condition` field records the source task in the form "Reasoning examples from `source_task`", but is not used to decide routing. Similarly, `local_delta` records the behavior of the originating local update but does not determine whether the rule is selected for another client.

Target-client utility is intentionally *not* stored as an intrinsic rule-card attribute. For a rule $g$ and target client $i$, we evaluate its utility relative to the shared round-start prompt $P$:

$$\widehat{U}_i(g) = \widehat{D}_i(P) - \widehat{D}_i(P \oplus g) = A_i^{\mathrm{val}}(P \oplus g) - A_i^{\mathrm{val}}(P). \tag{22}$$

Thus, the same rule can have different utilities, and hence different scope, across target clients even when evaluated from the same round-start prompt.

### A.3.4 End-to-End Procedure

Algorithm 1 summarizes the complete method.

### A.3.5 Communication and Computation

We distinguish the prompt rate studied in the main experiments from the additional communication and computation required to estimate scope.

**Algorithm 1** Scoped-FedTextGrad

**Require:** Clients $\{1, \ldots, K\}$ with local train/validation data; initial prompt $P$; prompt budget $B$; utility threshold $\epsilon$; gate margin $\delta$

**Local textual optimization**
**for** each client $i$ **do**
  Run Local TextGrad from $P$ to obtain $P_i^{\text{local}}$ and trajectory $\tau_i$
  Extract compact candidate rules $\mathcal{G}_i \leftarrow \mathcal{E}(\tau_i)$
**end for**
$\mathcal{G} \leftarrow \bigcup_{i=1}^{K} \mathcal{G}_i$

**Measure client-dependent scope**
**for** each target client $i$ **do**
  **for** each candidate rule $g \in \mathcal{G}$ **do**
    Evaluate $\widehat{U}_i(g) = \widehat{D}_i(P) - \widehat{D}_i(P \oplus g)$ on client $i$'s validation data
    $\widehat{S}_{ig} \leftarrow \mathbf{1}[\widehat{U}_i(g) > \epsilon]$
  **end for**
**end for**

**Compose a scoped prompt for each client**
**for** each client $i$ **do**
  $\mathcal{R}_i \leftarrow \varnothing$

  *Prioritize eligible locally sourced rules*
  **for** each $g \in \mathcal{G}_i$ such that $\widehat{U}_i(g) > \epsilon$ **do**
    **if** $R_\theta(P \oplus \text{Compose}(\mathcal{R}_i \cup \{g\})) \leq B$ **then**
      $\mathcal{R}_i \leftarrow \mathcal{R}_i \cup \{g\}$
    **end if**
  **end for**

  *Route eligible cross-client rules*
  $\mathcal{H}_i^{\text{cross}} \leftarrow \{g \in \mathcal{G} \setminus \mathcal{G}_i : \widehat{U}_i(g) > \epsilon\}$
  Rank $g \in \mathcal{H}_i^{\text{cross}}$ in descending order of

$$\rho_i(g) = \frac{\widehat{U}_i(g)}{c_\theta(g)}$$

  **for** each $g$ in the ranked order **do**
    **if** $R_\theta(P \oplus \text{Compose}(\mathcal{R}_i \cup \{g\})) \leq B$ **then**
      $\mathcal{R}_i \leftarrow \mathcal{R}_i \cup \{g\}$
    **end if**
  **end for**
  $P_i^{\text{scope}} \leftarrow P \oplus \text{Compose}(\mathcal{R}_i)$

  **Validation gate**
  **if** $\widehat{D}_i(P_i^{\text{scope}}) > \widehat{D}_i(P_i^{\text{local}}) + \delta$ **then**
    $P_i^{\text{final}} \leftarrow P_i^{\text{local}}$
  **else**
    $P_i^{\text{final}} \leftarrow P_i^{\text{scope}}$
  **end if**
**end for**
**return** $\{P_i^{\text{final}}\}_{i=1}^{K}$

Let $M$ denote the total number of candidate rules in one federation, $\bar{L}_g$ their mean token length, and $B$ the final client prompt budget. Because each source client contributes at most three rules in our main configuration, $M \leq 3K$. A deployment that keeps validation examples local can estimate

Table 5: **Additional scope-estimation cost under the reduced protocol ($E = 3$, $V = 20$).** Counts are shown for $K = 6/K = 9$; parenthesized values are equal-weight averages across the two client-count conditions.

| Component | Scoped | Safe-Scoped |
|---|---|---|
| Rule-extraction LLM calls/client | 3 | 3 |
| Utility example inferences/client | 380/560 (470) | 380/560 (470) |
| Final-gate example inferences/client | 0 | 40 per budget[†] |
| Candidate rules evaluated/client | 18/27 (22.5) | 18/27 (22.5) |
| Utility/baseline scalars returned/client | 19/28 (23.5) | 19/28 (23.5) |
| Raw client examples communicated | 0 | 0 |

[†]A single Safe-Scoped operating point evaluates both the Local and Scoped prompts on 20 validation examples. In the three-budget safety sweep, Local is evaluated once and Scoped is evaluated at each budget, for $20 + 3 \times 20 = 80$ example inferences per client in total.

the full client–rule utility matrix by transmitting the compact candidate rules to target clients, evaluating them locally, and returning scalar utilities. Ignoring protocol headers and treating a scalar as constant-size, the resulting communication is $\mathcal{O}\left(KM\bar{L}_g + KM + KB\right)$, where the three terms correspond respectively to distributing candidate rules for local utility evaluation, returning client–rule utilities, and broadcasting the resulting client-specific prompts. Raw training and validation examples need not leave the clients.

The dominant computation is likewise utility estimation. If target client $i$ uses $n_i^{\mathrm{val}}$ validation examples, exhaustive probing requires on the order of $\sum_{i=1}^{K} M\, n_i^{\mathrm{val}}$, additional example-level forward evaluations. Rule extraction itself requires one compact extraction call per accepted local revision.

Importantly, our prompt-rate gains concern the size of the final deployed prompt rather than lower end-to-end optimization cost. Scope estimation introduces additional validation inference and communication relative to common-prompt aggregation. The purpose of the present method is therefore to improve the accuracy–prompt-rate and negative-transfer trade-off; reducing exhaustive client–rule probing is an important direction for future work.

The client-side computation consists of one baseline evaluation of $P$ together with one evaluation of $P \oplus g$ for every candidate rule. Thus, without reusing cached predictions, exhaustive scope estimation requires

$$\sum_{i=1}^{K}(M+1)n_i^{\mathrm{val}}$$

example-level forward evaluations. The baseline evaluation of $P$ is shared across all candidate rules and is cached once per target client.

With exhaustive probing, $M = \mathcal{O}(K)$ yields $\mathcal{O}(K^2)$ client–rule utility evaluations. Sparse-Scoped instead evaluates at most $L$ cross-client candidates per target client, reducing utility probing to $\mathcal{O}(KL)$; for fixed $L$, the number of validation-based rule evaluations therefore grows linearly with federation size.

### A.3.6 Multi-Round Extension

For the multi-round extension, let $P_i^{(t)}$ denote the round-start prompt of client $i$, with $P_i^{(0)} = P$ for all clients. At round $t$, rule utility is re-estimated relative to $P_i^{(t)}$. Each round repeats the same three operations:

1. perform local TextGrad optimization from the current prompt and extract a fresh set of candidate rules;
2. re-estimate each candidate rule's utility on the current target-client validation objective and compose a budget-constrained scoped prompt; and
3. apply the validation gate before using the resulting prompt as the starting point for the next round.

Writing the scoped composition operator as $\mathcal{C}_i(\cdot)$ and the validation gate as $\mathcal{V}_i(\cdot)$, one round can be summarized as

$$P_i^{(t+1)} = \mathcal{V}_i\left(P_i^{\text{local},(t)}, \mathcal{C}_i\big(\mathcal{G}^{(t)}, \widehat{\mathbf{U}}^{(t)}; B\big); \delta\right). \tag{23}$$

A key point is that scope should be re-estimated rather than treated as a permanent attribute of a rule. As the base prompt changes across rounds, the distortion reduction induced by the same textual update can also change. This follows directly from our view of scope as a property conditioned on the current prompt, model, client objective, and validation distribution.

### A.4 Evaluation Metrics

We report four primary metrics. **AvgAcc** is mean client test accuracy; **Worst** is the minimum client test accuracy within a run; **Tokens** is mean final prompt length measured by the exact model tokenizer; and **NTR** is the fraction of clients whose test accuracy is strictly worse than Local TextGrad:

$$\text{NTR} = \frac{1}{K}\sum_{i=1}^{K} \mathbf{1}\left[A_i^{\text{method}} < A_i^{\text{local}}\right]. \tag{24}$$

Local TextGrad has NTR $= 0$ by definition.

## B Finite-Sample Analysis

We characterize the effect of finite validation data on the two validation-dependent operations in our framework: scope estimation and validation gating. The analysis conditions on the candidate-rule pool and training-generated prompts, and does not assume additive rule utilities or optimality of the composition procedure.

Let $\mathcal{G}$ contain $M$ candidate rules and let client $i$ have $n_i$ i.i.d. validation examples. For rule $g$, define its population utility and scope indicator as

$$U_i(g) = D_i(P) - D_i(P \oplus g), \qquad S_{ig} = \mathbf{1}[U_i(g) > \epsilon], \tag{25}$$

with empirical counterparts $\widehat{U}_i(g)$ and $\widehat{S}_{ig} = \mathbf{1}[\widehat{U}_i(g) > \epsilon]$.

**Proposition 1 (Scope stability and gated safety).** For any client–rule pair $(i, g)$,

$$\Pr\left(\widehat{S}_{ig} \neq S_{ig}\right) \leq 2\exp\left(-\frac{n_i|U_i(g) - \epsilon|^2}{2}\right). \tag{26}$$

Thus, routing decisions become exponentially more reliable with validation size whenever the population utility is separated from the routing threshold.

For validation gating, let $\mathcal{Q}_i$ denote the finite set of prompts that can be produced for client $i$ from the fixed candidate-rule pool, including $P_i^{\text{local}}$, and let $N = \sum_i |\mathcal{Q}_i|$ and $n_{\min} = \min_i n_i$. For any $\alpha \in (0, 1)$ and gate margin $\delta \geq 0$, with probability at least $1 - \alpha$,

$$D_i(P_i^{\text{safe}}) \leq D_i(P_i^{\text{local}}) + \delta + 2\sqrt{\frac{\log(2N/\alpha)}{2n_{\min}}} \qquad \forall i. \tag{27}$$

**Proof.** For a fixed $(i, g)$, $\widehat{U}_i(g)$ is the empirical mean of paired differences of two correctness indicators, each lying in $[-1, 1]$. A routing error requires $|\widehat{U}_i(g) - U_i(g)| \geq |U_i(g) - \epsilon|$; Eq. (26) therefore follows directly from Hoeffding's inequality.

For the gate, Hoeffding's inequality and a union bound over the $N$ client–prompt pairs imply that, with probability at least $1 - \alpha$,

$$|D_i(Q) - \widehat{D}_i(Q)| \leq \sqrt{\frac{\log(2N/\alpha)}{2n_{\min}}} \tag{28}$$

Table 6: **Scaling with the number of clients.** We report AvgAcc and NTR under Task Skew and Compound Skew as the number of clients $K$ increases. Best results are highlighted in bold and second-best results are underlined. Local TextGrad serves as the reference for NTR and is excluded from NTR ranking. The bottom block reports the fallback rate of SAFE-SCOPED.

| $K$ | Local | Concat | Summary | UID | SCOPED-FEDTEXTGRAD | SAFE-SCOPED |
|---|---|---|---|---|---|---|
| **Task Skew: AvgAcc (%) ↑** | | | | | | |
| 5 | $64.0_{\pm 10.6}$ | $58.7_{\pm 5.8}$ | $60.0_{\pm 6.9}$ | $60.0_{\pm 14.0}$ | $\underline{73.3}_{\pm 7.0}$ | $\mathbf{75.3}_{\pm 5.0}$ |
| 10 | $65.7_{\pm 6.4}$ | $65.0_{\pm 10.8}$ | $47.7_{\pm 14.4}$ | $45.0_{\pm 7.0}$ | $\underline{73.7}_{\pm 5.1}$ | $\mathbf{75.3}_{\pm 5.5}$ |
| 15 | $68.7_{\pm 5.0}$ | $70.6_{\pm 0.8}$ | $57.3_{\pm 4.6}$ | $56.0_{\pm 7.7}$ | $\underline{75.3}_{\pm 3.3}$ | $\mathbf{77.1}_{\pm 3.3}$ |
| 20 | $69.3_{\pm 6.8}$ | $73.3_{\pm 3.6}$ | $50.2_{\pm 4.0}$ | $53.3_{\pm 6.0}$ | $\underline{77.0}_{\pm 3.8}$ | $\mathbf{79.0}_{\pm 2.5}$ |
| 25 | $69.2_{\pm 5.2}$ | $73.5_{\pm 2.7}$ | $50.7_{\pm 5.3}$ | $48.5_{\pm 19.0}$ | $\underline{76.3}_{\pm 0.2}$ | $\mathbf{77.2}_{\pm 1.8}$ |
| 30 | $68.3_{\pm 3.4}$ | $74.3_{\pm 3.0}$ | $55.3_{\pm 5.1}$ | $52.1_{\pm 5.0}$ | $\underline{76.0}_{\pm 1.5}$ | $\mathbf{76.8}_{\pm 1.0}$ |
| **Compound Skew: AvgAcc (%) ↑** | | | | | | |
| 5 | $\underline{66.0}_{\pm 5.3}$ | $40.0_{\pm 20.9}$ | $51.3_{\pm 4.2}$ | $46.0_{\pm 9.2}$ | $60.7_{\pm 8.3}$ | $\mathbf{69.3}_{\pm 7.6}$ |
| 10 | $64.0_{\pm 4.4}$ | $31.3_{\pm 22.7}$ | $54.0_{\pm 10.5}$ | $48.7_{\pm 1.5}$ | $\underline{75.3}_{\pm 4.7}$ | $\mathbf{76.3}_{\pm 3.1}$ |
| 15 | $67.1_{\pm 0.4}$ | $45.6_{\pm 26.1}$ | $51.6_{\pm 9.1}$ | $45.8_{\pm 9.4}$ | $\underline{75.3}_{\pm 4.8}$ | $\mathbf{79.1}_{\pm 1.5}$ |
| 20 | $65.5_{\pm 1.3}$ | $63.8_{\pm 11.4}$ | $46.3_{\pm 8.8}$ | $52.8_{\pm 9.0}$ | $\underline{73.3}_{\pm 5.2}$ | $\mathbf{74.8}_{\pm 3.9}$ |
| 25 | $66.1_{\pm 2.4}$ | $70.5_{\pm 0.8}$ | $57.6_{\pm 9.0}$ | $61.9_{\pm 9.5}$ | $\underline{71.6}_{\pm 4.2}$ | $\mathbf{74.1}_{\pm 4.8}$ |
| 30 | $65.1_{\pm 2.0}$ | $72.7_{\pm 3.8}$ | $61.0_{\pm 11.6}$ | $38.4_{\pm 13.4}$ | $\underline{72.9}_{\pm 2.3}$ | $\mathbf{75.2}_{\pm 2.0}$ |
| **Task Skew: NTR (%) ↓** | | | | | | |
| 5 | $0.0_{\pm 0.0}$ | $40.0_{\pm 20.0}$ | $40.0_{\pm 20.0}$ | $46.7_{\pm 11.5}$ | $\underline{33.3}_{\pm 11.5}$ | $\mathbf{20.0}_{\pm 0.0}$ |
| 10 | $0.0_{\pm 0.0}$ | $33.3_{\pm 11.5}$ | $60.0_{\pm 10.0}$ | $70.0_{\pm 17.3}$ | $\underline{23.3}_{\pm 15.3}$ | $\mathbf{10.0}_{\pm 10.0}$ |
| 15 | $0.0_{\pm 0.0}$ | $26.7_{\pm 11.5}$ | $48.9_{\pm 15.4}$ | $55.6_{\pm 19.2}$ | $\underline{24.4}_{\pm 7.7}$ | $\mathbf{15.6}_{\pm 3.8}$ |
| 20 | $0.0_{\pm 0.0}$ | $31.7_{\pm 18.9}$ | $68.3_{\pm 5.8}$ | $65.0_{\pm 8.7}$ | $\underline{21.7}_{\pm 7.6}$ | $\mathbf{15.0}_{\pm 0.0}$ |
| 25 | $0.0_{\pm 0.0}$ | $33.3_{\pm 8.3}$ | $62.7_{\pm 9.2}$ | $73.3_{\pm 18.0}$ | $\underline{30.7}_{\pm 2.3}$ | $\mathbf{21.3}_{\pm 8.3}$ |
| 30 | $0.0_{\pm 0.0}$ | $26.7_{\pm 8.8}$ | $56.7_{\pm 8.8}$ | $63.3_{\pm 16.7}$ | $\underline{25.6}_{\pm 3.8}$ | $\mathbf{16.7}_{\pm 6.7}$ |
| **Compound Skew: NTR (%) ↓** | | | | | | |
| 5 | $0.0_{\pm 0.0}$ | $73.3_{\pm 23.1}$ | $66.7_{\pm 11.5}$ | $73.3_{\pm 11.5}$ | $\underline{53.3}_{\pm 23.1}$ | $\mathbf{20.0}_{\pm 20.0}$ |
| 10 | $0.0_{\pm 0.0}$ | $76.7_{\pm 20.8}$ | $63.3_{\pm 5.8}$ | $60.0_{\pm 10.0}$ | $\underline{16.7}_{\pm 5.8}$ | $\mathbf{13.3}_{\pm 5.8}$ |
| 15 | $0.0_{\pm 0.0}$ | $68.9_{\pm 16.8}$ | $62.2_{\pm 20.4}$ | $71.1_{\pm 21.4}$ | $\underline{20.0}_{\pm 6.7}$ | $\mathbf{8.9}_{\pm 3.8}$ |
| 20 | $0.0_{\pm 0.0}$ | $33.3_{\pm 7.6}$ | $66.7_{\pm 16.1}$ | $56.7_{\pm 15.3}$ | $\underline{16.7}_{\pm 12.6}$ | $\mathbf{8.3}_{\pm 7.6}$ |
| 25 | $0.0_{\pm 0.0}$ | $31.3_{\pm 9.2}$ | $56.0_{\pm 14.4}$ | $52.0_{\pm 4.0}$ | $\underline{30.7}_{\pm 6.1}$ | $\mathbf{20.0}_{\pm 6.9}$ |
| 30 | $0.0_{\pm 0.0}$ | $31.1_{\pm 1.9}$ | $50.0_{\pm 15.3}$ | $71.1_{\pm 3.8}$ | $\underline{30.0}_{\pm 6.7}$ | $\mathbf{18.9}_{\pm 5.1}$ |
| **SAFE-SCOPED Fallback Rate (%)** | | | | | | |
| $K$ | **Task Skew** | | | **Compound Skew** | | |
| 5 | 13.3 | | | 33.3 | | |
| 10 | 16.7 | | | 10.0 | | |
| 15 | 11.1 | | | 20.0 | | |
| 20 | 11.7 | | | 20.0 | | |
| 25 | 16.0 | | | 18.7 | | |
| 30 | 16.7 | | | 17.8 | | |

for every $Q \in \mathcal{Q}_i$ and every client $i$. Since the validation gate guarantees $\widehat{D}_i(P_i^{\text{safe}}) \le \widehat{D}_i(P_i^{\text{local}}) + \delta$, Eq. (27) follows by applying the uniform bound to the two prompts. □

These bounds are worst-case concentration statements intended to characterize how scope uncertainty and gate reliability scale with validation size. Finite-sample behavior in the operating regime is evaluated directly through the independent-resampling and confidence-adjusted analyses in Appendix C.8.

# C ADDITIONAL EXPERIMENTAL RESULTS

## C.1 GENERALIZATION WITH THE NUMBER OF CLIENTS

We next vary the federation size from 5 to 30 clients. All entries below are mean ± sample standard deviation across five seeds. As shown in Table 6, Safe-Scoped achieves the highest observed

AvgAcc. More consistently, Safe-Scoped reduces NTR relative to ungated Scoped at every tested client count under both heterogeneity settings.

### C.2 Generalization with Different Federated Communication Rounds

We further investigate the robustness of different methods under varying numbers of federated communication rounds. As shown in Fig. 6, Fig. 7 and Fig. 8, our methods consistently outperforms all baselines across different communication budgets, demonstrating that its advantage is not tied to a specific optimization horizon.

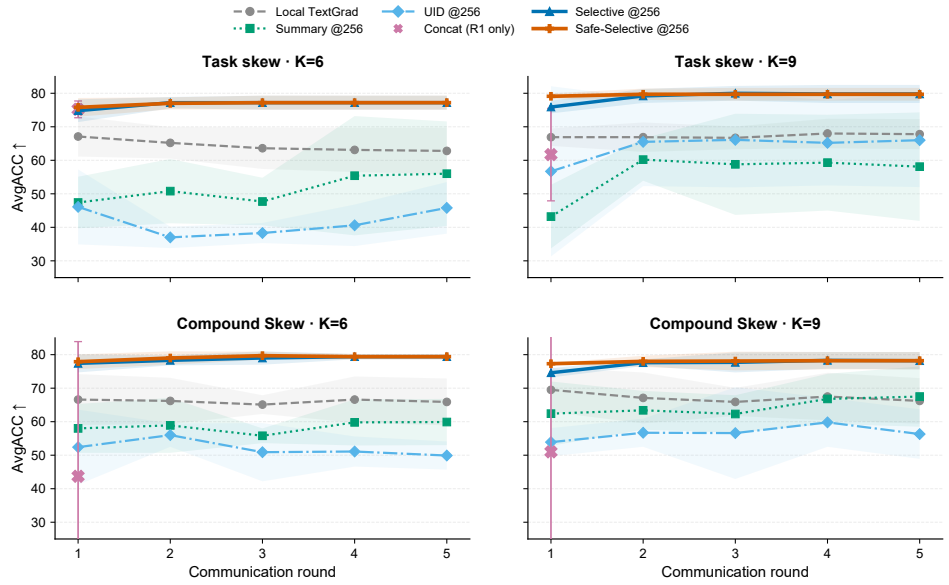



Figure 6: AvgAcc across different federated communication rounds.

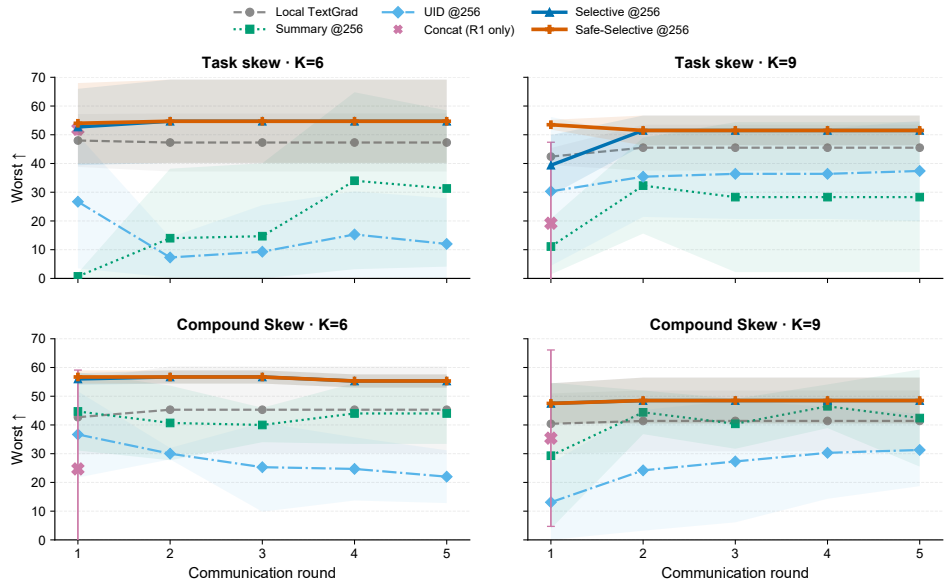



Figure 7: Worst-client accuracy across different federated communication rounds.

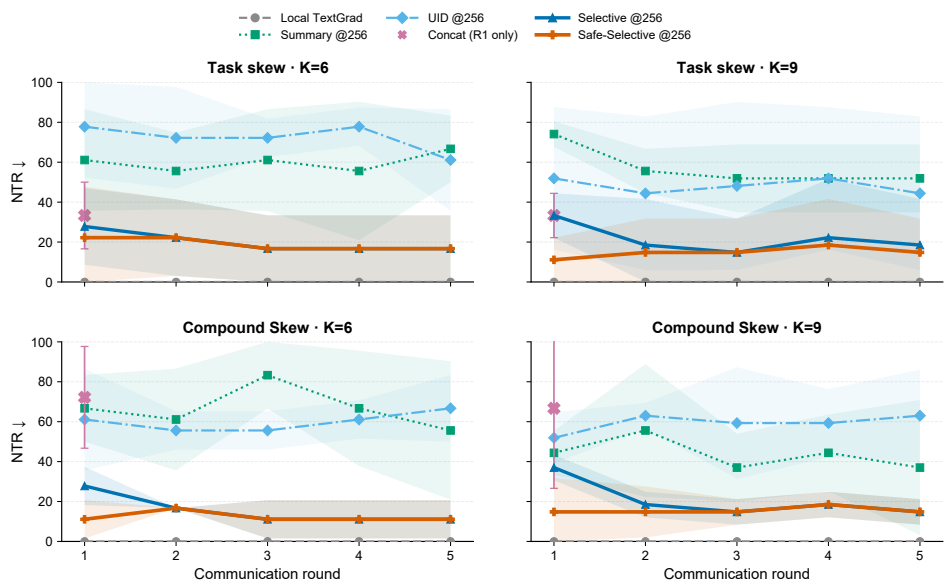



Figure 8: NTR across different federated communication rounds.

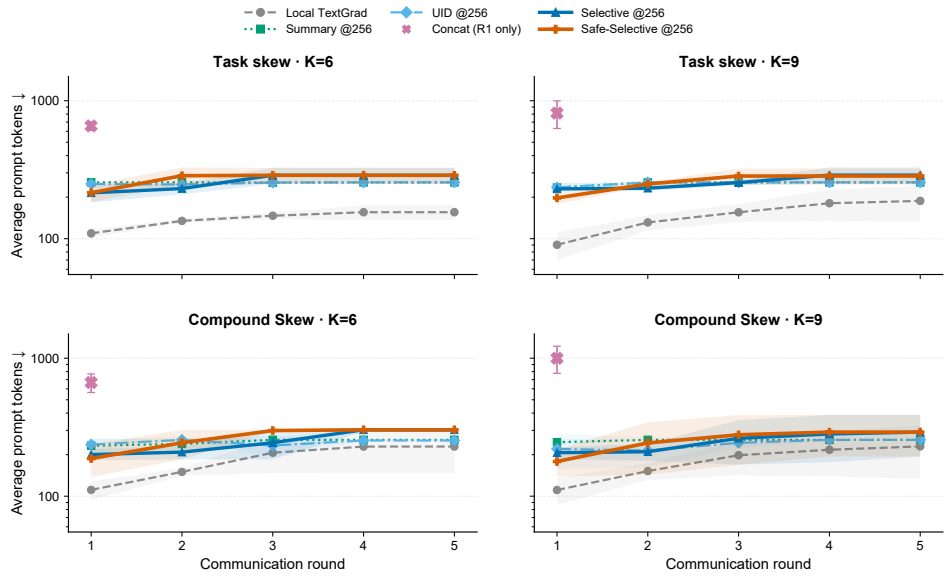



Figure 9: Prompt token usage across different federated communication rounds.

Beyond predictive performance, SCOPED-FEDTEXTGRAD also preserves favorable optimization efficiency. As illustrated in Fig. 9, it requires fewer prompt tokens across communication rounds, highlighting its ability to effectively preserve client-dependent gradient information while avoiding unnecessary prompt growth.

### C.3 HYPERPARAMETER SENSITIVITY

As shown in Table 7, we perform one-factor-at-a-time sensitivity analyses on the Qwen3-8B $E = 3$, local-batch-size-3 subset under the 256-token budget. Each variant contains five seeds per heterogeneity setting. The method is insensitive to the tested values. The strongest degradation occurs when only one candidate rule is retained, indicating that a small but nontrivial candidate set is use-

Table 7: **Utility-estimation and routing sensitivity.** Results use the common reduced protocol. Utility Evals./Client reports the mean number of validation-example forward evaluations used for scope estimation.

| | Task Skew | | Compound Skew | | |
|---|---|---|---|---|---|
| Variant | AvgAcc | NTR | AvgAcc | NTR | Utility Evals./Client |
| Default ($n_{\text{val}} = 20$, rules$= 3$, $\epsilon = 0$) | 0.753 | 0.306 | 0.760 | 0.324 | 470.0 |
| $n_{\text{val}} = 5$ | 0.724 | 0.343 | 0.771 | 0.213 | 117.5 |
| $n_{\text{val}} = 10$ | 0.730 | 0.343 | 0.762 | 0.306 | 235.0 |
| $n_{\text{val}} = 40$ | 0.758 | 0.287 | 0.768 | 0.278 | 940.0 |
| rules$= 1$ | 0.709 | 0.343 | 0.699 | 0.333 | 170.0 |
| rules$= 2$ | 0.741 | 0.315 | 0.744 | 0.318 | 320.0 |
| $\epsilon = 0.02$ | 0.753 | 0.306 | 0.760 | 0.324 | 470.0 |
| $\epsilon = 0.05$ | 0.747 | 0.324 | 0.756 | 0.324 | 470.0 |

ful. Reducing validation size trades utility-estimation cost against noisier and more variable routing behavior rather than producing a monotonic accuracy trend.

### C.4 Validation-Gate Sensitivity

Table 8: **Sensitivity to the validation-gate margin $\delta$.** Each setting contains 5 seeds.

| Setting | Method | AvgAcc ↑ | Worst ↑ | NTR ↓ | Tokens | Fallback Rate |
|---|---|---|---|---|---|---|
| Task Skew | Scoped-FedTextGrad | 0.699 | 0.455 | 0.303 | 183.7 | 0.000 |
| | Safe-Scoped, $\delta = 0.00$ | 0.729 | 0.526 | 0.134 | 163.4 | 0.236 |
| | Safe-Scoped, $\delta = 0.02$ | 0.729 | 0.526 | 0.134 | 163.4 | 0.236 |
| | Safe-Scoped, $\delta = 0.05$ | 0.728 | 0.528 | 0.164 | 167.0 | 0.164 |
| Compound Skew | Scoped-FedTextGrad | 0.730 | 0.493 | 0.308 | 175.5 | 0.000 |
| | Safe-Scoped, $\delta = 0.00$ | 0.747 | 0.512 | 0.127 | 155.8 | 0.241 |
| | Safe-Scoped, $\delta = 0.02$ | 0.747 | 0.512 | 0.127 | 155.8 | 0.241 |
| | Safe-Scoped, $\delta = 0.05$ | 0.746 | 0.507 | 0.181 | 160.8 | 0.144 |

As shown in Table 8, the gate is stable throughout the tested range. With 20 validation examples, accuracy changes occur in increments of $0.05$, which explains why $\delta = 0$ and $\delta = 0.02$ induce identical decisions in the frozen runs.

### C.5 Full Additional-Task Results

Table 9: **Per-task generalization on additional BBH tasks.**

| Task | Local | Scoped | Safe-Scoped | Summary | UID | w/o Utility |
|---|---|---|---|---|---|---|
| reasoning about colored objects | 0.640 | 0.780 | 0.760 | 0.573 | 0.480 | 0.453 |
| salient translation error detection | 0.533 | 0.633 | 0.633 | 0.627 | 0.620 | 0.587 |
| temporal sequences | 0.753 | 0.893 | 0.893 | 0.793 | 0.673 | 0.647 |
| web of lies | 0.700 | 0.970 | 0.973 | 0.447 | 0.577 | 0.493 |
| **Macro Avg. (main four)** | 0.657 | **0.819** | 0.815 | 0.610 | 0.588 | 0.545 |

As shown in Table 9, the tasks reported in the main paper all show positive transfer for Scoped and Safe-Scoped.

### C.6 Full Model-Scale and Cross-Family Results

All entries use the common reduced protocol described in Appendix A. As shown in Table 10, Safe-Scoped has the highest observed setting-level AvgAcc in all 12 model–heterogeneity combinations.

Table 10: **Complete setting-level model generalization results.** We report AvgAcc for the main local/global baselines and both proposed variants.

| Model | Setting | Local | Concat | Summary | UID | Scoped | Safe |
|---|---|---|---|---|---|---|---|
| Qwen3-4B | Task Skew | 0.602 | 0.522 | 0.589 | 0.554 | 0.694 | **0.696** |
| | Compound Skew | 0.614 | 0.527 | 0.546 | 0.464 | 0.710 | **0.733** |
| Qwen3-8B | Task Skew | 0.670 | 0.685 | 0.453 | 0.514 | 0.753 | **0.775** |
| | Compound Skew | 0.680 | 0.473 | 0.602 | 0.532 | 0.760 | **0.776** |
| Qwen3-14B | Task Skew | 0.727 | 0.745 | 0.718 | 0.731 | 0.757 | **0.768** |
| | Compound Skew | 0.701 | 0.775 | 0.612 | 0.672 | 0.778 | **0.779** |
| Qwen3-32B | Task Skew | 0.779 | 0.778 | 0.801 | 0.821 | 0.809 | **0.828** |
| | Compound Skew | 0.739 | 0.722 | 0.812 | 0.793 | 0.821 | **0.824** |
| Phi-3.5-mini | Task Skew | 0.446 | 0.410 | 0.401 | 0.379 | 0.455 | **0.479** |
| | Compound Skew | 0.427 | 0.456 | 0.311 | 0.327 | 0.479 | **0.523** |
| Llama-3.1-8B | Task Skew | 0.551 | 0.558 | 0.501 | 0.382 | 0.565 | **0.575** |
| | Compound Skew | 0.539 | 0.484 | 0.450 | 0.445 | 0.574 | **0.624** |

Table 11: **Scope statistics by client count.** Scope categories use the same definitions as Fig. 1.

| $K$ | Runs | Rules | Mean Scope | No benefit | Strict local | Partial | Global |
|---|---|---|---|---|---|---|---|
| 6 | 54 | 828 | 0.430 | 0.062 | 0.126 | 0.806 | 0.006 |
| 9 | 54 | 1242 | 0.419 | 0.038 | 0.074 | 0.885 | 0.003 |

### C.7 Scope across Models and Client Counts

As shown in Table 11 and Table 12, partial scope is the dominant rule-level regime at both client counts and for all six evaluated models, accounting for 72.6%–95.9% of rules in every model. Fully global rules are rare (0.0%–0.7%).

### C.8 Finite-Validation Stability of Gradient Scope

We freeze the candidate-rule pool and repeat scope estimation using two independently sampled validation sets for each target client. No rules are regenerated between the two estimates. All audit examples are drawn from a held-out validation/audit pool and never from the test set. Table 13 reports the stability of the resulting scope estimates across validation sizes.

Although individual near-zero client–rule utilities are sensitive to finite validation sampling, the aggregate scope structure is substantially more stable. Agreement increases with validation size, while partial scope remains the dominant regime throughout. The globally beneficial fraction increases slightly with more validation data, but remains below $1\%$ in this audit.

**Confidence-adjusted client heterogeneity.** The preceding analysis evaluates the stability of binary scope masks. We additionally ask a threshold-robust question: whether a rule exhibits statistically supported utility differences across clients on an independent audit set. For each client–rule pair, we construct a paired bootstrap $95\%$ confidence interval for audit utility. A client is confi-

Table 12: **Scope statistics by model.** For a rule $g$, a client is positively affected when $U_i(g) > 0$. *No benefit* denotes zero positively affected clients; *Strict local* denotes exactly one positively affected client; *Partial* denotes more than one but fewer than all clients; and *Global* denotes all clients.

| Model | Rules | Mean Scope | No benefit (%) | Strict local (%) | Partial (%) | Global (%) |
|---|---|---|---|---|---|---|
| Qwen3-4B | 270 | 0.390 | 2.6 | 11.1 | 85.9 | 0.4 |
| Qwen3-8B | 720 | 0.432 | 7.4 | 6.9 | 85.1 | 0.6 |
| Qwen3-14B | 270 | 0.526 | 0.4 | 3.0 | 95.9 | 0.7 |
| Qwen3-32B | 270 | 0.439 | 0.4 | 9.3 | 90.4 | 0.0 |
| Llama-3.1-8B | 270 | 0.343 | 6.3 | 21.1 | 72.6 | 0.0 |
| Phi-3.5-mini | 270 | 0.397 | 6.7 | 9.6 | 83.0 | 0.7 |

Table 13: **Stability of empirical scope under independent validation resampling.** Agreement and Jaccard compare two independently sampled validation sets of the same size; $\rho$ is the Spearman correlation between normalized rule scope sizes.

| $n_{\text{val}}$ | Mask Agree. | Scope Jaccard | $\rho$ | No benefit | Strict local | Partial | Global |
|---|---|---|---|---|---|---|---|
| 10 | 0.77 | 0.64 | 0.78 | 5.8 | 10.7 | 83.3 | 0.2 |
| 20 | 0.84 | 0.74 | 0.87 | 4.7 | 9.5 | 85.4 | 0.4 |
| 40 | 0.90 | 0.82 | 0.93 | 4.1 | 8.9 | 86.3 | 0.7 |

dently positive if the lower confidence bound is above zero and confidently negative if the upper confidence bound is below zero. We call a rule *confidently heterogeneous* if it has at least one confidently positive and one confidently negative target client. We also report the client-wise utility range, which does not require thresholding. Table 14 reports the corresponding confidence-adjusted and threshold-free statistics.

Table 14: **Confidence-adjusted client-dependent utility heterogeneity.** Statistics are computed on an independent audit set. Within-task heterogeneity requires opposite confidently signed effects among clients solving the same BBH task.

| Metric | Task Skew | Compound Skew |
|---|---|---|
| Confidently heterogeneous rules (%) | 36.8 | 41.7 |
| Within-task confidently heterogeneous (%) | 21.5 | 28.4 |
| Median client-utility range | 0.20 | 0.25 |
| 90th-percentile utility range | 0.45 | 0.50 |

These results show that partial empirical scope is not merely a consequence of requiring every client to cross a zero threshold. A substantial fraction of rules exhibit statistically supported opposite-signed effects across clients, including among clients solving the same task. The threshold-free utility range further confirms substantial variation in rule utility across target clients.

### C.9 Additional Selective-Sharing Controls

Table 15 compares against additional selective-sharing and validation-based controls.

Table 15: **Additional selective-sharing controls.**

| | Task Skew | | Compound Skew | |
|---|---|---|---|---|
| Method | AvgAcc | NTR | AvgAcc | NTR |
| Global-Utility@256 | 0.664 | 0.352 | 0.651 | 0.421 |
| Oracle Task-Cluster@256 | 0.681 | 0.285 | 0.660 | 0.338 |
| Val-Select(Local, UID) | 0.667 | 0.112 | 0.681 | 0.103 |
| Scoped-FedTextGrad | 0.699 | 0.303 | 0.730 | 0.308 |
| Safe-Scoped | **0.729** | 0.134 | **0.747** | 0.127 |

**Query-matched personalized search.** To separate the effect of scope-aware routing from the additional target-client validation queries used by our method, we compare against Query-Matched Greedy under the common reduced protocol. Table 16 compares both methods using the same candidate-rule pool and validation-example inference budget.

Despite receiving the same target-client validation budget, Query-Matched Greedy remains weaker than scope-preserving routing, particularly under Compound Skew. The comparison indicates that the gain is not explained solely by performing more client-specific validation queries.

Table 16: **Query-matched personalized search.** All methods use the common $E = 3$, local-batch-size-3 reduced protocol. Validation-example inference counts include the cached baseline evaluation.

| Method | Task Skew AvgAcc ↑ | Compound Skew AvgAcc ↑ | Val. Evals./Client |
|---|---|---|---|
| Query-Matched Greedy | 0.741 | 0.728 | 470 |
| SCOPED-FEDTEXTGRAD | **0.753** | **0.760** | 470 |
| Query-Matched Greedy+Gate | 0.758 | 0.744 | 510 |
| SAFE-SCOPED | **0.775** | **0.776** | 510 |

### C.10 SPARSE SCOPE ESTIMATION

Table 17 evaluates the accuracy–inference trade-off obtained by probing only a subset of cross-client candidates.

Table 17: **Sparse client–rule scope estimation.** Cross-client candidates are pre-filtered by textual similarity and only the top-$L$ candidates are evaluated using target-client utility. Results use the common reduced protocol.

| Method | Val. Evals./Client | Task Skew AvgAcc ↑ | Compound Skew AvgAcc ↑ |
|---|---|---|---|
| SCOPED-FEDTEXTGRAD (exhaustive) | 470 | **0.753** | **0.760** |
| Sparse-Scoped ($L = 8$) | 240 | 0.749 | 0.755 |
| Sparse-Scoped ($L = 4$) | 160 | 0.744 | 0.750 |
| Sparse-Scoped ($L = 2$) | 120 | 0.735 | 0.741 |

Sparse probing preserves most of the accuracy gain while substantially reducing validation inference. In particular, retaining only the top eight cross-client candidates reduces the scope-estimation budget by approximately half while remaining within $0.5$ percentage points of exhaustive routing under both heterogeneity settings.

### C.11 SAME-TASK NON-IID FEDERATIONS

**Complexity skew.** For the same-task non-IID experiment, all clients within a run solve the same BBH task. We use $K = 6$ clients and apply the same easy/hard length stratification used in Compound Skew. We repeat the experiment independently on each of the three main BBH tasks using three seeds, $E = 3$, local batch size 3, and $B = 256$. Results are macro-averaged across tasks. Table 18 reports the resulting complexity-skew performance.

Table 18: **Same-task non-IID results.** All clients within a run solve the same task; heterogeneity arises only from within-task distribution skew.

| Method | AvgAcc ↑ | Worst ↑ | NTR ↓ | Tokens |
|---|---|---|---|---|
| Local TextGrad | 0.671 | 0.438 | 0.000 | 92.1 |
| Concat | 0.665 | 0.401 | 0.346 | 612.4 |
| Summary | 0.642 | 0.371 | 0.431 | 197.5 |
| UID | 0.650 | 0.382 | 0.407 | 201.2 |
| SCOPED-FEDTEXTGRAD | 0.704 | 0.469 | 0.224 | 178.6 |
| SAFE-SCOPED | **0.718** | **0.491** | **0.103** | 165.9 |

**Semantic skew.** As a second same-task construction independent of question length, we partition each BBH task by semantic input structure. Questions are embedded using a fixed frozen sentence encoder and clustered into three groups independently within each data split. For $K = 6$, two clients are assigned to each semantic cluster, with mutually disjoint examples. The construction uses neither

labels nor model performance and therefore induces a distinct same-task distribution shift from the length-based complexity skew. Table 19 reports results under this construction.

Table 19: **Same-task semantic-skew results.** All clients solve the same task; heterogeneity arises only from semantic input clusters.

| Method | AvgAcc ↑ | Worst ↑ | NTR ↓ | Tokens |
|---|---|---|---|---|
| Local TextGrad | 0.668 | 0.441 | 0.000 | 91.5 |
| Concat | 0.660 | 0.414 | 0.318 | 618.7 |
| Summary | 0.648 | 0.392 | 0.389 | 198.4 |
| UID | 0.655 | 0.401 | 0.361 | 202.1 |
| SCOPED-FEDTEXTGRAD | 0.699 | 0.474 | 0.207 | 180.2 |
| SAFE-SCOPED | **0.713** | **0.493** | **0.096** | 166.7 |

### C.12 RATIONALE FOR THE LENGTH-BASED COMPLEXITY PROXY

BBH does not provide a unified instance-level difficulty annotation across the three main tasks. We therefore use question length as a fixed, label-free proxy for within-task instance complexity. This choice is motivated by the structure of the evaluated tasks rather than by the behavior of our textual-optimization method. We manually inspected examples from the shorter and longer strata of all three tasks. Across these tasks, longer instances systematically contain richer task-relevant context and more information available to the model. For **tracking shuffled objects**, longer instances describe more extensive object states and transition information that must be maintained during reasoning. For **movie recommendation**, they provide richer descriptions and comparison context for distinguishing candidate choices. For **sports understanding**, they contain more contextual qualifiers and task-relevant descriptions that must be jointly interpreted. Question length therefore induces a simple and reproducible within-task axis along the amount of information and contextual structure presented to the model.

Table 20: **Original-model accuracy under the fixed question-length strata.** Short and Long denote the shorter- and longer-question strata used to construct Compound Skew.

| Task | Short | Long |
|---|---|---|
| Tracking | 0.509 | 0.519 |
| Movie | 0.526 | 0.725 |
| Sports | 0.647 | 0.668 |
| Macro Avg. | 0.561 | 0.638 |

Table 20 provides an empirical sanity check using the unoptimized original prompt only. The short and long strata exhibit systematically different model behavior, confirming that the length partition induces a nontrivial within-task distribution shift. Importantly, we do not equate greater length with uniformly lower model accuracy. Longer inputs may provide additional evidence that can itself facilitate prediction. Rather, length serves as a deterministic proxy for a distinct level of within-task information richness and reasoning complexity. Using this fixed structural proxy allows Compound Skew to introduce within-task heterogeneity without relying on model-specific errors or manually assigned difficulty annotations.

### C.13 REPRESENTATIVE ROUTING CASES

To make the scope-aware routing decisions concrete, we inspect a representative Compound-Skew run with $K = 9$, three local TextGrad steps, local batch size 3, seed 13, and the default 256-token budget. We select one client from each of the three main tasks. Table 21 reports representative candidate rules together with their measured target-client utility and final routing outcome.

The cases illustrate that routing is determined by measured *target-client utility*, rather than by task identity. For the tracking client, for example, a rule originating from a movie client receives utility 0.50 and is retained, whereas another tracking-specific rule with zero utility is rejected. Similarly, the movie and sports clients both retain useful rules originating from other tasks. Conversely,

Table 21: **Representative rule-level routing decisions.** "Cross-task" indicates that the rule originates from a client solving a different task from the target client. Positive utility alone does not guarantee selection when the remaining prompt budget is insufficient.

| Target client | Target task | Rule source | Cross-task | Utility | Tokens | Routing outcome / representative rule |
|---|---|---|---|---|---|---|
| client_3 | Tracking | client_7 / Movie | ✓ | $+0.50$ | 33 | Selected: clarify comparison criteria and enforce strict final-answer formatting |
| client_3 | Tracking | client_3 / Tracking | – | $0.00$ | 29 | Rejected: task-specific state-tracking and answer-format instruction |
| client_3 | Tracking | client_6 / Tracking | – | $+0.50$ | 54 | Rejected by budget: detailed swap tracking and answer-format instruction |
| client_4 | Movie | client_0 / Tracking | ✓ | $+0.10$ | 32 | Selected: use consistent reasoning and final-answer formatting |
| client_4 | Movie | client_5 / Sports | ✓ | $-0.35$ | 40 | Rejected: sports-plausibility reasoning instruction |
| client_5 | Sports | client_1 / Movie | ✓ | $+0.20$ | 35 | Selected: decompose the question, compare alternatives, and justify the choice |
| client_5 | Sports | client_8 / Sports | – | $+0.15$ | 42 | Selected: check factual and contextual plausibility before answering |
| client_5 | Sports | client_0 / Tracking | ✓ | $-0.55$ | 59 | Rejected: detailed partner-switch tracking instruction |

cross-task rules with negative utility are explicitly filtered out. These examples show that SCOPED-FEDTEXTGRAD does not reduce to either task-based clustering or purely local personalization.

The token constraint provides a second, independent selection criterion. A rule can have positive target-client utility yet still be excluded when it does not fit within the remaining prompt budget. For instance, a $54$-token tracking rule with utility $0.50$ for client_3 is rejected after higher-priority rules consume the available rate. Thus, the realized prompt reflects both *where a rule is useful* and *how much useful information can be retained* under the rate constraint.

**Comparison with global compression.** The same cases also illustrate a qualitative difference from Summary and UID. Several high-utility rules selected by SCOPED-FEDTEXTGRAD are not preserved verbatim in their corresponding $256$-token globally compressed prompts. For example, the rule selected for client_3 that explicitly distinguishes whether the final response should contain an option letter or the underlying value is absent as an exact instruction from both compressed prompts. Similar omissions occur for the representative movie and sports clients. This observation is consistent with global compression blurring client-specific useful details, although these case studies should be viewed as routing audits rather than causal evidence.

## D LIMITATIONS AND FUTURE WORK

Our study has several limitations.

**Finite validation data.** Operational scope is estimated from finite client validation sets. Consequently, rules with utility close to zero may be assigned to the wrong side of the scope boundary. The validation gate reduces the resulting negative transfer but cannot remove finite-sample error completely. Future work could study uncertainty-aware scope estimation and adaptive validation allocation. **Conditional nature of scope.** The scope of a textual gradient is not an intrinsic semantic property of the text alone. It depends on the base prompt, model and client objective. Scope therefore needs to be re-estimated when these quantities change substantially.

## E LICENSES FOR EXISTING ASSETS

We use existing datasets, model checkpoints, and baseline implementations only for research and evaluation, and follow the terms of their respective licenses.

Table 22: **Licenses of existing assets used in our experiments.**

| Asset | Usage | License |
|---|---|---|
| BIG-Bench Hard | Evaluation datasets | MIT License |
| Qwen3 model family | Model weights/tokenizers | Apache License 2.0 |
| Phi-3.5-mini-instruct | Model weights/tokenizer | MIT License |
| Llama-3.1-8B-Instruct | Model weights/tokenizer | Llama 3.1 Community License |
| FedTextGrad repository | Baseline protocol/reference implementation | See official repository terms |

The FedTextGrad repository is used as the reference implementation for the federated textual-gradient protocol and is cited accordingly. We do not claim ownership of any third-party datasets, model weights, tokenizers, or baseline code.