DIALECT: CONTINUOUS-SPACE COMMUNICATION WITHIN AND ACROSS LANGUAGE MODELS

Anonymous authors
Paper under double-blind review

ABSTRACT

Multi-agent systems (MAS) enable large language models (LLMs) to collaborate on complex tasks by combining complementary capabilities and exchanging intermediate reasoning. MAS primarily communicate through natural language, which introduces latency through message generation and limits the information passed to the receiver by selecting discrete tokens from distributions over possible continuations. Continuous representations offer an alternative communication medium, but differences in tokenization, embedding dimensionality, and representational structure pose challenges to cross-model communication. We introduce **D**istributional **I**nter-**A**gent **L**atent **E**xchange via **C**ross-model **T**ranslation (DIALECT), a framework for continuous-space communication between agents using the same or different LLMs. The source agent constructs continuous thoughts as stochastic mixtures of token embeddings, retaining contributions from multiple continuations at each step. For same-model communication, thoughts transfer directly; for cross-model communication, a bridge trained in two stages maps them into the target's input embedding space, with both LLMs frozen. We evaluate DIALECT in a two-agent planning-and-solving setting on 13 benchmarks across mathematical reasoning, knowledge question answering, and code generation across four models from three families, spanning 3B–8B. DIALECT significantly outperforms single-agent targets, same-model baselines, and cross-model baselines. Compared to text transfer constrained to equal source generation steps, it achieves significantly higher accuracy with lower latency. Against full-text communication, it maintains statistically equivalent accuracy in all model pairs, while delivering a median speedup of $2.02\times$. These results support stochastic soft-token generation and a learned cross-model bridge as a practical approach to continuous-space communication within and across LLMs.

1 INTRODUCTION

Large language models (LLMs) are increasingly deployed in multi-agent systems (MAS), where agents with different roles, complementary skills, and access to different tools collaborate toward the same set of goals (Li et al., 2023; Hong et al., 2024; Wu et al., 2023). By combining skills and exchanging intermediate solutions, agents build on one another's strengths, using iterative feedback to examine competing perspectives and revise answers for better reasoning (Liang et al., 2024; Chen et al., 2024) and factual accuracy (Du et al., 2023). Heterogeneous LLMs further combine complementary model capabilities for stronger responses (Wang et al., 2025). These motivate examining how agents communicate and how the communication medium affects accuracy and efficiency.

Natural language serves as the typical medium for agent communication, allowing models to exchange information without requiring compatible internal representations. However, natural-language communication incurs substantial computational overhead, as the source model must generate each message sequentially, and the receiving model must tokenize and process that message before producing its response (Ramesh & Li, 2025). Beyond latency, collapsing rich predictive distributions into discrete tokens acts as an information bottleneck, discarding uncertainty and alternative candidate continuations that could guide the receiver (Pham et al., 2024). These limitations have motivated investigating continuous representations as an alternative medium for inter-agent communication (Pham et al., 2024; Ramesh & Li, 2025; Zou et al., 2025). While such representations can transfer directly when agents share the same model, extending continuous communication

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.

to different LLMs introduces an additional challenge. Disparate embedding dimensions and representational structure prevent one model's continuous message from being transferred directly to another. Cross-model continuous communication therefore requires not only generating informative thoughts, but also adapting them into representations the receiving model can process.

A parallel appears in the human brain. If specialized brain areas are viewed as distinct agents, they do not coordinate using symbolic language; extensive work in neuroscience shows that many forms of complex reasoning operate independently of the brain's language network (Fedorenko & Varley, 2016; Benn et al., 2023; Fedorenko et al., 2024). Consistent with this, nonlinguistic continuous formats, such as cognitive maps, serve as rich internal media for reasoning (Yousif & Epstein, 2026). At the circuit level, cortical areas interact through a low-dimensional "communication subspace", where source areas route signals along target-specific projections (Semedo et al., 2019). This biological principle motivates the view that MAS do not need to verbalize thoughts into text or expose their entire internal state; instead, agents could communicate through continuous space, with learned, targeted representational projections adapting thoughts to the receiving model's representation space.

Drawing on these insights, we introduce **D**istributional **I**nter-**A**gent **L**atent **E**xchange via **C**ross-model **T**ranslation (DIALECT), a framework for continuous-space communication among agents using the same or different LLMs. DIALECT combines two components: (i) training-free stochastic soft-token generation, adapted from single-agent continuous reasoning (Wu et al., 2026), which builds continuous thoughts from mixtures of candidate token embeddings rather than single tokens; and (ii) a learned cross-model bridge, which maps thoughts into the target's input embedding space when agents differ in tokenizer, embedding dimension, and representational structure. We train the bridge first on paired source and target embeddings, then end-to-end against receiver predictions, keeping both LLMs frozen. The source model passes its continuous thought to the receiving model directly when both agents use the same model, and through the bridge otherwise; the receiving model then uses this thought together with the original task to generate its response.

We evaluate DIALECT in a two-agent planning-and-solving setting across mathematical reasoning, knowledge-based question answering, and code generation, using models from the Qwen, Gemma, and Mistral families. In this work, (i) we evaluate the effectiveness of stochastic soft-token continuous thoughts when source and target agents use the same model. We compare DIALECT with standalone target models, existing same-model continuous communication baselines (CIPHER, Pham et al., 2024; and LatentMAS, Zou et al., 2025), budget-matched text transfer, and full-text generated plans. (ii) We examine whether the learned bridge transfers useful continuous thoughts between different models within and across model families. We compare DIALECT with standalone target models, training-free and cache-based baselines, budget-matched text transfer, and full-text generated plans. (iii) Finally, we vary the communication budget within DIALECT to examine how the length of continuous thoughts affects accuracy and inference time. Overall, DIALECT establishes a new efficiency frontier for inter-agent communication and significantly outperforms prior continuous baselines and budget-matched text transfer. Compared with full-text communication, it retains over 98% of benchmark-averaged accuracy across all model pairs while achieving a median speedup of 2.02×, with paired equivalence tests supporting statistically equivalent accuracy in all pairs.

2 Related Work

Multi-Agent Language-Model Collaboration. MAS organize collaboration through interactions from role-based dialogue (Li et al., 2023) to structured workflows that assign agents responsibilities such as design, implementation, and testing (Hong et al., 2024; Qian et al., 2024). Programmable conversation frameworks support these interactions among agents, tools, and human participants (Wu et al., 2023). Agents can use one another's responses to refine a solution, either by reviewing others' reasoning and revising their answers (Du et al., 2023) or combine responses from different model families through successive stages of generation and aggregation (Wang et al., 2025). These systems use natural language as the communication medium; DIALECT similarly separates responsibilities across agents but replaces the text channel with continuous representations.

Continuous Reasoning and Latent Communication. Most MAS communicate through text, which requires sequential message generation and repeated processing by receiving agents. Ramesh & Li (2025) identify both the inference cost and the information lost when internal states are reduced to text. These concerns motivate methods for reasoning and communication through con-

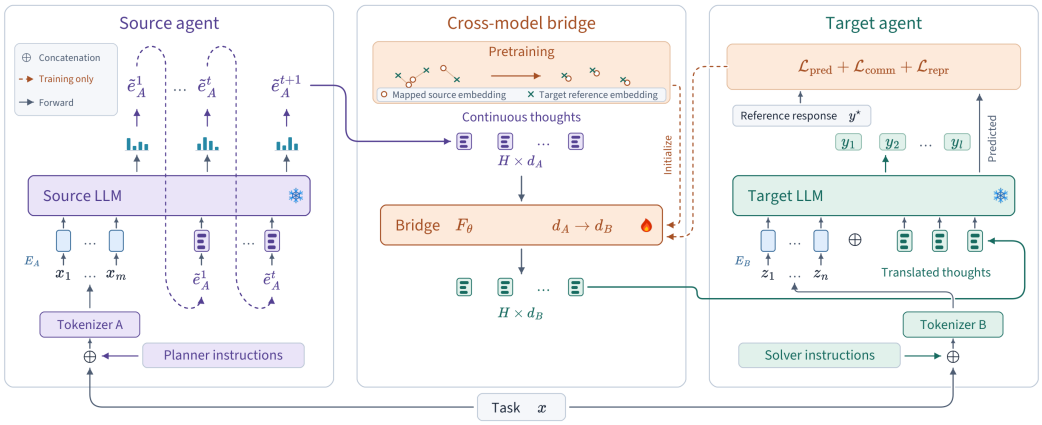


Figure 1: Overview of DIALECT. Source generates H thoughts and transfers via F_θ to guide target's answer. Bridge is pretrained on paired embeddings and refined with both LLMs frozen.

tinuous space. Within a single model, COCONUT feeds hidden states back as inputs (Hao et al., 2024), while CODI uses self-distillation to learn continuous reasoning (Shen et al., 2025). Other methods form inputs from probability-weighted token embeddings (Zhang et al., 2026) or combine a sampled token with the predicted distribution (Zhuang et al., 2026). However, including several token contributions does not guarantee diverse reasoning. Wu et al. (2026) find that high-probability tokens can dominate deterministic mixtures and introduce Dirichlet and Gumbel-Softmax sampling to reduce this effect. Continuous representations can also guide frozen models through task-specific prefixes (Li & Liang, 2021) or assistant-generated thoughts, with extensions that encourage more diverse thoughts for the same input (Xu et al., 2025a;b). In agent communication, continuous representations carry information between models through token embeddings and intermediate activations (Pham et al., 2024; Ramesh & Li, 2025), or through learned prefixes conveying information from other agents (Zheng et al., 2026). LatentMAS combines hidden-state reasoning with shared KV-cache memory (Zou et al., 2025), while Interlat and RecursiveMAS learn connections that support latent communication between agents (Du et al., 2026; Zou et al., 2026). DIALECT builds on stochastic soft token generation (Wu et al., 2026) as the communication medium and, unlike methods that require the same model, supports communication between agents with different LLMs.

Representation Alignment. Continuous communication between different models requires connecting their representation spaces. Model stitching studies mappings between independently trained components (Lenc & Vedaldi, 2015; Bansal et al., 2021), while anchor-based establish correspondence between representations through shared reference samples (Moschella et al., 2022; Maiorca et al., 2023). For LLMs, learned mappings enable transferring features, probes, and steering vectors (Chen et al., 2026a), and shared interfaces let analysis tools and intermediate states be reused across models (Kim et al., 2026). Representation alignment also operates on KV caches, using learned transformations (Chen et al., 2026b), gated fusion (Fu et al., 2026), shared spaces (Dery et al., 2026), or joint memory from both agents' caches (Liu et al., 2026b). A complementary approach constructs receiver inputs from sender states; XBridge combines lexical anchors with hidden states to preserve entity identity across models (Yang et al., 2026). StateBridge aligns message hidden states with input embeddings without gradient-based training, evaluating agents that share model weights within each run (Peng et al., 2026). Most of these exchange KV caches or intermediate hidden states, requiring access to those internal representations. DIALECT instead transfers stochastic mixtures of source token embeddings into the target's input embedding space through a learned bridge, keeping both LLMs frozen without accessing their internal representations.

3 METHOD

We introduce DIALECT, a framework for continuous-space communication between frozen LLM agents without requiring intermediate hidden states or internal KV caches (Figure 1). Following

Wang et al. (2023), we separate planning and solution generation: the source agent constructs a continuous thought via stochastic soft-token generation (Wu et al., 2026), which the target agent uses to produce its response. For agents using the same model, the thought transfers directly; for different models, a learned cross-model bridge maps it into the target's input embedding space. We define the communication setting, describe continuous generation, bridge architecture, and training.

3.1 Problem Formulation

Let π_A and π_B denote source and target LLMs with vocabularies $\mathcal{V}_A$ and $\mathcal{V}_B$, input embedding dimensions d_A and d_B, and embedding tables $E_A = \{e_i^A\}_{i\in\mathcal{V}_A}$ and $E_B = \{e_j^B\}_{j\in\mathcal{V}_B}$. Both receive input x with role-specific instructions. Within an H-step communication budget, the source autoregressively generates $Z_A = (\tilde{e}_A^1, \ldots, \tilde{e}_A^H)$, where $\tilde{e}_A^t \in \mathbb{R}^{d_A}$ is the soft token at step t (Section 3.2).

We consider two settings: *same-model communication*, where both agents use the same checkpoint and DIALECT requires no additional training, and *cross-model communication*, where they use different checkpoints from the same or different model families. The target receives $Z_B = (\tilde{e}_B^1, \ldots, \tilde{e}_B^H)$, where $\tilde{e}_B^t \in \mathbb{R}^{d_B}$ is defined by

$$\tilde{e}_B^t = \begin{cases} \tilde{e}_A^t, & \text{same-model communication,} \\ F_\theta(\tilde{e}_A^t), & \text{cross-model communication,} \end{cases} \tag{1}$$

where $F_\theta : \mathbb{R}^{d_A} \to \mathbb{R}^{d_B}$ is a learned cross-model bridge with parameters θ (Section 3.3). The target receives Z_B as H additional input-embedding positions appended after its prompt.

3.2 Generation in Continuous Space

At each autoregressive step, an LLM assigns probabilities to plausible continuations. Conventional decoding selects one token, discarding alternatives; continuous generation constructs weighted token embeddings, retaining multiple candidates' contributions for subsequent computation and communication. Following Wu et al. (2026), we sample weights using Dirichlet for same-model communication, emphasizing a few continuations and passing soft tokens directly through the shared embedding space, and Gumbel–Softmax for cross-model communication, retaining broader contributions before mapping to the target's embedding space. For model π with vocabulary $\mathcal{V}$, let $E = \{e_i\}_{i\in\mathcal{V}}$ be its learned input embedding table. At step t, the model produces probabilities $P^t = (p_i^t)_{i\in\mathcal{V}}$. We apply top-$k$ and top-p truncation to retain a token set $\mathcal{S}^t$ and renormalize as $q_i^t = p_i^t / \sum_{j\in\mathcal{S}^t} p_j^t$ for $i \in \mathcal{S}^t$. From the resulting distribution $Q^t = (q_i^t)_{i\in\mathcal{S}^t}$, we sample mixture weights $W^t = (w_i^t)_{i\in\mathcal{S}^t}$ and construct the soft token as

$$\tilde{e}^t = c^t \sum_{i\in\mathcal{S}^t} w_i^t e_i, \tag{2}$$

where c^t rescales the Dirichlet mixture to the model's mean input embedding norm, compensating for the norm reduction from weighted averaging, and equals one for Gumbel–Softmax. The resulting soft token is fed back at step $t+1$. Sampling and norm rescaling are detailed in Section A.

3.3 Cross-Model Bridge

Continuous generation produces soft tokens in the source model's input embedding space. We introduce a cross-model bridge F_θ that maps each source soft token $\tilde{e}_A^t$ to the target's input embedding space $\tilde{e}_B^t = F_\theta(\tilde{e}_A^t)$. We train the bridge in two stages. Standalone pretraining learns a mapping from paired source and target representations, placing the bridge's outputs in the target's embedding space before task gradients are applied. End-to-end training then optimizes the bridge within the communication pipeline using the target model's predictions of reference responses. Both LLMs remain frozen; only the bridge parameters are updated.

Bridge architecture. The bridge maps a source soft token $u = \tilde{e}_A^t$ into the target's input embedding space by predicting its direction and magnitude separately. Let $\bar{u}$ denote the normalized source vector. To obtain the direction, we project $\bar{u}$ linearly to the target dimension, add a nonlinear adjustment produced by an MLP, and normalize the resulting vector. A second MLP takes $\bar{u}$ as input and predicts a scalar, which is transformed into a positive, bounded magnitude. Multiplying the normalized direction by this magnitude gives the mapped soft token. For target models that also require

per-layer embeddings (PLE), such as Gemma (Team et al., 2026), an additional head predicts these PLE. The target receives both the mapped soft token and the predicted PLE at each communication position. The full architecture and PLE construction are detailed in Section B.

Standalone pretraining. We pretrain the bridge on paired embeddings $\mathcal{D}_{\text{pair}} = \{(u_n, v_n)\}_{n=1}^{N}$, where u_n is a source embedding and v_n its target-space reference. When vocabulary entries and token IDs match, we construct pairs by applying the same source-sampled token weights to both models' input embedding tables. Otherwise, we pair input-token embeddings at corresponding sentence boundaries in shared reference responses, formatted and tokenized separately for each model. Pair construction and training data are detailed in sections C and 4.2. We align predicted and reference vectors in direction, magnitude, and relative squared error (RSE):

$$\mathcal{A}(a, b) = 1 - \cos(a, b) + \beta_n \left(\log \frac{\|a\|_2}{\|b\|_2} \right)^2 + \beta_r \frac{\|a - b\|_2^2}{\|b\|_2^2}, \tag{3}$$

where a and b denote the predicted and reference vectors, respectively, and β_n and β_r weight the magnitude and RSE terms. We minimize $N^{-1} \sum_{n=1}^{N} \mathcal{A}(F_\theta(u_n), v_n)$. For targets requiring PLE, we supervise main embeddings and per-layer residuals.

End-to-end training. After pretraining, we optimize the bridge within the source–target pipeline using reference responses containing reasoning and final answers. The objective is

$$\mathcal{L} = \mathcal{L}_{\text{pred}} + \mathcal{L}_{\text{comm}} + \mathcal{L}_{\text{repr}}. \tag{4}$$

The prediction loss $\mathcal{L}_{\text{pred}}$ encourages the target to generate the reference response from the continuous thought. We combine reference-response cross-entropy with selective distillation from target predictions conditioned on a source-generated text plan. We select the better of two independently sampled plans and apply distillation only if its reference-response cross-entropy is lower than that under the continuous thought. The communication loss $\mathcal{L}_{\text{comm}}$ encourages the target to benefit from input-specific information in the continuous thought. We combine comparisons against shuffled and zero-valued messages with a penalty for increased response loss relative to receiving no thought (diagnostics in Section D.1). The representation loss $\mathcal{L}_{\text{repr}}$ combines embedding alignment (Equation (3)) with a penalty on output norms above model-specific bounds. Details in Sections B and D.

4 Experimental Setup

4.1 Models

We evaluate Qwen3-8B, Qwen3-4B (Yang et al., 2025), Ministral-3-3B-Reasoning-2512 (Liu et al., 2026a), and gemma-4-E4B-it (Team et al., 2026), with thinking enabled for all (Section I). Same-model setting uses Qwen3-8B as source and target. Cross-model settings are Qwen3-8B $\rightarrow$ Qwen3-4B, Qwen3-8B $\rightarrow$ Ministral-3B, Qwen3-8B $\rightarrow$ Gemma-E4B, and Gemma-E4B $\rightarrow$ Ministral-3B, testing whether smaller targets benefit from larger sources within and across model families.

4.2 Training Data

We train bridges on the math, science, and code splits of Llama-Nemotron-Post-Training-Dataset (Bercovich et al., 2025) for mathematical reasoning, knowledge tasks, and code generation, respectively, with a separate bridge for each source–target pair, domain, and communication budget H.

4.3 Baselines

Single-agent and text transfer baselines. For same-model and cross-model communication, we evaluate each target independently without another agent's message, assessing whether communicated thoughts improve on the target's own reasoning. Two text-transfer baselines use source-generated natural-language plans to guide target responses. (i) Budget-matched text transfer limits source plans to the same generation steps as continuous communication. (ii) Full-text transfer allows the full generation budget, providing a higher-cost baseline to assess whether continuous communication approaches its accuracy at lower latency.

Same-model continuous baselines. For same-model communication, we adapt CIPHER (Pham et al., 2024) and LatentMAS (Zou et al., 2025) within our evaluation pipeline. CIPHER generates soft tokens as probability-weighted combinations of input embeddings. We adapt its multiagent debate setting to a source–target setting. Our LatentMAS adaptation generates latent thoughts through repeated hidden-state feedback and transfers the mapped representations to the target (Section L).

Cross-model baselines. We compare two training-free baselines: Training-free, uses DIALECT's stochastic soft-token generation, and CIPHER-Cross, our cross-model adaptation of CIPHER (Pham et al., 2024), uses the full-vocabulary softmax distribution. Both transfer source token weights through exact matches in the target vocabulary, renormalize the matched weights, and reconstruct soft tokens using target embeddings. Neither requires paired training data or parameter updates, but both discard unmatched-token contributions (Section E). We also adapt Cache-to-Cache (C2C; Fu et al., 2026), which learns to project and fuse source key–value caches into the target (Section N).

4.4 Evaluation

We evaluate mathematical reasoning, multiple-choice knowledge, and Python code generation. The math benchmarks are AIME 2024 (Hugging Face H4, 2024), AIME 2025 (Zhang & Math-AI Team, 2025), AMC 2023 (Math-AI, 2023), MATH-500 (Hendrycks et al., 2021; Lightman et al., 2024), and GSM8K (Cobbe et al., 2021). The knowledge benchmarks are OpenBookQA (Mihaylov et al., 2018), GPQA Diamond (Rein et al., 2023), ARC-Challenge (Clark et al., 2018), ARC-Easy (Clark et al., 2018), and MedQA (Jin et al., 2020). The code benchmarks are HumanEval (Chen et al., 2021), MBPP (Austin et al., 2021), and MBPP+ (Liu et al., 2023). See section F for more details.

We evaluate all models and baselines identically over three seeds, reporting mean accuracy and standard deviation. For math and knowledge, responses are correct if either rule-based checking or GPT-5.2 judging accepts them. Rule evaluators extract answers and check reference equivalence for math, or extract and compare option labels for knowledge. The judge receives the processed response and reference answer without the original question, using Section G templates. Code correctness is determined by executable tests without LLM judging. We report single-sample pass@1, the fraction of problems whose generated solution passes all required tests.

We compare budgets $H \in \{40, 80, 160\}$ and report mean inference time per sample (Section K). At $H = 80$, one-sided Wilcoxon signed-rank tests assess accuracy gains and latency reductions across 13 benchmarks paired by benchmark and seed, with Holm correction within each pair and outcome ($\alpha = .05$). We report geometric-mean speedups and accuracy retention, DIALECT's benchmark-averaged accuracy as a percentage of full-text's, averaging seeds per benchmark. We assess accuracy equivalence to full text using paired Wilcoxon-based TOST within ± 2 percentage points (Section P). For budget analysis, we fit a seed-level mixed-effects model with logit-transformed accuracy, log latency centered within each pair–benchmark, and a random intercept (Section R). One-sided Wilcoxon signed-rank tests compare adjacent budgets, paired by benchmark and seed per pair, with Holm correction across the four accuracy and latency comparisons (Section S).

4.5 Implementation Details

Each agent receives a role-specific system instruction and the same task input in its model's chat template. Following Wang et al. (2023), the source develops a plan, which the target uses to produce a boxed answer for math, an option label for knowledge tasks, or a complete Python solution for code generation. Exact instructions are in section H. Unless specified, continuous thoughts contain $H = 80$ embeddings. Continuous source generation uses logit temperature 0.6, top-$k = 30$, top-$p = 0.95$, Dirichlet concentration $\alpha = 4$, and Gumbel–Softmax temperature $\tau = 0.5$. Autoregressive sampling uses temperature 0.6, top-$k = 20$, and top-$p = 0.95$, with a 16,384-token limit and end-of-sequence stopping. Bridge optimization and hardware details are in Sections D and J.

5 Results

We evaluate accuracy and latency in same- and cross-model communication. We first compare our method with the baselines described in Section 4.3 at the default budget $H = 80$. We then examine how varying the budget over $H \in \{40, 80, 160\}$ affects the accuracy–latency trade-off.

Table 1: Same-model communication accuracy ($M \pm SD$, 3 seeds) and latency (s/sample).

Method	Metric	AIME 2024	AIME 2025	AMC 2023	MATH 500	GSM 8K	Open Book	GPQA Diamond	ARC C	ARC E	Med QA	Human Eval	MBPP	MBPP +
Single	Acc.	$66.7_{\pm 3.3}$	$50.0_{\pm 5.8}$	$85.0_{\pm 4.3}$	$95.7_{\pm 1.4}$	$95.0_{\pm 0.8}$	$94.2_{\pm 1.2}$	$58.9_{\pm 1.8}$	$95.8_{\pm 0.9}$	$98.4_{\pm 0.6}$	$80.9_{\pm 0.8}$	$88.4_{\pm 2.1}$	$85.3_{\pm 1.8}$	$78.8_{\pm 1.5}$
Text (80)	Acc.	**70.0**$_{\pm 3.3}$	**70.0**$_{\pm 5.8}$	$90.0_{\pm 4.3}$	$96.0_{\pm 1.3}$	$95.5_{\pm 0.6}$	$94.2_{\pm 1.4}$	$58.1_{\pm 1.8}$	$95.4_{\pm 0.9}$	$98.0_{\pm 0.4}$	$81.1_{\pm 0.7}$	$87.8_{\pm 2.2}$	$84.8_{\pm 1.7}$	$80.2_{\pm 1.5}$
	Latency	244.0	269.8	201.0	101.9	35.9	30.0	195.6	36.4	19.8	82.5	75.8	50.1	83.1
CIPHER	Acc.	$63.3_{\pm 3.3}$	$63.3_{\pm 5.8}$	$87.5_{\pm 5.0}$	$96.0_{\pm 1.4}$	$94.4_{\pm 0.7}$	$95.4_{\pm 1.1}$	$55.9_{\pm 1.3}$	$95.8_{\pm 0.9}$	$83.8_{\pm 0.5}$	$81.0_{\pm 0.7}$	**91.5**$_{\pm 2.2}$	**93.0**$_{\pm 1.7}$	$80.7_{\pm 1.4}$
	Latency	241.1	217.5	172.3	**80.2**	29.7	26.1	133.5	**29.6**	17.1	55.9	53.7	46.4	80.3
LatentMAS	Acc.	$40.0_{\pm 3.3}$	$50.0_{\pm 5.8}$	$80.0_{\pm 4.3}$	$81.8_{\pm 1.2}$	$86.0_{\pm 0.7}$	$85.2_{\pm 1.3}$	$49.0_{\pm 1.7}$	$83.4_{\pm 0.8}$	$83.8_{\pm 0.6}$	$73.6_{\pm 0.7}$	$50.6_{\pm 2.2}$	$52.5_{\pm 1.4}$	$40.2_{\pm 1.7}$
	Latency	287.5	253.6	231.1	163.1	63.1	25.5	163.5	34.1	17.8	79.2	66.1	51.1	81.2
DIALECT	Acc.	**70.0**$_{\pm 3.3}$	$66.7_{\pm 3.3}$	**92.5**$_{\pm 2.5}$	**96.6**$_{\pm 0.9}$	**95.7**$_{\pm 0.5}$	**95.8**$_{\pm 0.9}$	**60.6**$_{\pm 1.3}$	**96.4**$_{\pm 0.7}$	**98.5**$_{\pm 0.5}$	**81.4**$_{\pm 0.5}$	$91.3_{\pm 1.4}$	$92.6_{\pm 1.9}$	**81.2**$_{\pm 1.4}$
	Latency	**216.3**	**202.1**	**164.1**	81.9	**22.8**	**19.7**	**125.8**	33.0	**15.9**	**50.0**	**47.6**	**39.0**	**76.7**
Full text	Acc.	$76.7_{\pm 3.3}$	$73.3_{\pm 5.8}$	$92.5_{\pm 4.3}$	$97.0_{\pm 1.3}$	$96.2_{\pm 0.8}$	$95.4_{\pm 1.4}$	$61.1_{\pm 1.3}$	$96.1_{\pm 0.6}$	$98.6_{\pm 0.6}$	$81.7_{\pm 0.6}$	$92.7_{\pm 2.2}$	$93.3_{\pm 1.6}$	$81.7_{\pm 1.5}$
	Latency	562.9	499.3	370.4	208.1	46.3	44.0	285.0	58.2	31.7	132.1	106.2	89.6	151.5

5.1 Same-model Communication

Table 1 shows results with Qwen3-8B as both source and target. DIALECT improves over the single-agent target on all benchmarks, spanning mathematical reasoning, knowledge tasks, and code generation. The mean accuracy improvement across benchmarks is 3.54 percentage points, with a statistically significant paired comparison ($p < .001$). These results suggest that the target benefits from receiving the source's continuous thought when both agents use the same checkpoint.

Against budget-matched text transfer, DIALECT matches or exceeds accuracy on 12 benchmarks and lowers latency on all, with a mean accuracy gain of 1.40 percentage points ($p < .001$) and geometric-mean speedup of $1.34\times$ ($p < .001$). Budget-matched text improves over the single-agent target on some benchmarks but matches or underperforms it on others. Thus, continuous thoughts reduce latency under the same source generation budget, with higher accuracy on benchmarks.

Against CIPHER (Pham et al., 2024), DIALECT achieves higher mean accuracy on 11 benchmarks and lower mean latency on 11 benchmarks, improving both on nine. The mean accuracy improvement is 2.89 percentage points ($p < .001$) and a geometric-mean speedup of $1.10\times$ ($p < .001$). Against LatentMAS (Zou et al., 2025), it improves on all benchmarks with a mean accuracy gain of 20.24 percentage points ($p < .001$) and geometric-mean speedup of $1.40\times$ ($p < .001$). Qualitative analysis of LatentMAS shows repetitive decoded thoughts and uninformative reconstructions, suggesting a communication procedure limitation (Section O).

Compared with full-text, DIALECT reduces latency across all benchmarks, achieving a geometric-mean speedup of $2.24\times$ ($p < .001$) with a mean accuracy drop of 1.31 percentage points (98.51% retention). Wilcoxon-based TOST supports equivalence within ± 2 percentage points for the center of paired accuracy differences ($p = .0032$), showing short continuous thoughts retain much of communication's benefit at lower latency.

5.2 Cross-model Communication

Tables 2 and 4 show results of four pairs within and across model families. Below, results are in the order Qwen3-8B $\rightarrow$ Qwen3-4B, Qwen3-8B $\rightarrow$ Gemma, Qwen3-8B $\rightarrow$ Ministral, and Gemma $\rightarrow$ Ministral. Compared with the single-agent target, DIALECT improves accuracy on all 13 benchmarks for every pair. The mean accuracy improvements are 2.69, 5.49, 2.66, and 1.73 percentage points, respectively (all $p < .001$). The paired comparisons are statistically significant for all pairs. These gains suggest that the benefits of continuous thoughts extend beyond agents using the same model, and the learned bridge enables targets to use thoughts generated by different models.

Against Training-free baseline, DIALECT matches or exceeds accuracy in nearly all settings while reducing overall latency. Mean accuracy improvements are 1.24, 9.81, 0.91, and 21.77 percentage points, respectively (all $p < .001$), with geometric-mean speedups of $1.03\times$, $1.07\times$, $1.08\times$, and $1.26\times$, respectively (all $p < .001$). Against CIPHER-Cross, DIALECT achieves higher mean accuracy and lower mean latency on all benchmarks for each pair, with mean accuracy gains of 3.84, 11.64, 2.79, and 23.50 percentage points, respectively (all $p < .001$) and geometric-mean speedups

Table 2: Cross-model communication across four source–target pairs: accuracy ($M \pm SD$, 3 seeds) and latency (s/sample).

Method	Metric	AIME 2024	AIME 2025	AMC 2023	MATH 500	GSM 8K	Open Book	GPQA Diamond	ARC C	ARC E	Med QA	Human Eval	MBPP	MBPP +
Qwen3-8B → Qwen3-4B														
Single	Acc.	66.7$_{\pm 5.8}$	53.3$_{\pm 5.8}$	85.0$_{\pm 5.0}$	94.8$_{\pm 1.0}$	94.7$_{\pm 0.8}$	94.6$_{\pm 1.3}$	56.1$_{\pm 2.2}$	95.0$_{\pm 0.9}$	97.4$_{\pm 0.5}$	73.2$_{\pm 0.7}$	89.0$_{\pm 1.8}$	85.6$_{\pm 1.8}$	78.5$_{\pm 1.5}$
Text (80)	Acc.	60.0$_{\pm 3.3}$	43.3$_{\pm 3.3}$	85.0$_{\pm 4.3}$	93.8$_{\pm 1.1}$	94.0$_{\pm 0.6}$	94.2$_{\pm 1.3}$	55.1$_{\pm 2.0}$	94.6$_{\pm 0.6}$	97.1$_{\pm 0.6}$	72.9$_{\pm 0.9}$	88.4$_{\pm 2.1}$	85.2$_{\pm 1.9}$	77.8$_{\pm 1.6}$
	Latency	372.6	265.0	107.5	95.4	35.0	11.0	192.7	37.7	20.5	80.8	71.7	42.7	84.7
Training-free	Acc.	**73.3**$_{\pm 5.8}$	56.7$_{\pm 5.8}$	95.0$_{\pm 5.0}$	95.0$_{\pm 1.3}$	94.9$_{\pm 0.7}$	95.0$_{\pm 1.1}$	**58.9**$_{\pm 2.3}$	94.8$_{\pm 0.5}$	97.1$_{\pm 0.5}$	72.9$_{\pm 0.5}$	90.9$_{\pm 1.8}$	79.1$_{\pm 2.0}$	79.1$_{\pm 1.7}$
	Latency	309.4	210.4	101.0	**76.9**	26.1	6.0	140.7	26.4	16.6	**57.2**	**50.0**	40.0	77.1
DIALECT	Acc.	**73.3**$_{\pm 3.3}$	**60.0**$_{\pm 3.3}$	**97.5**$_{\pm 2.5}$	**96.0**$_{\pm 0.8}$	**95.5**$_{\pm 0.5}$	**95.4**$_{\pm 0.8}$	56.7$_{\pm 2.8}$	**95.3**$_{\pm 0.7}$	**97.7**$_{\pm 0.4}$	**73.9**$_{\pm 0.4}$	**91.3**$_{\pm 1.4}$	**86.4**$_{\pm 1.8}$	**79.9**$_{\pm 1.2}$
	Latency	**286.1**	216.0	**98.1**	77.2	**25.3**	**5.7**	140.4	**24.3**	**16.3**	**57.2**	**50.0**	**38.4**	**75.6**
Full text	Acc.	70.0$_{\pm 5.8}$	70.0$_{\pm 5.8}$	92.5$_{\pm 4.3}$	97.0$_{\pm 1.1}$	95.1$_{\pm 0.8}$	95.8$_{\pm 1.1}$	58.9$_{\pm 2.1}$	95.9$_{\pm 0.7}$	98.3$_{\pm 0.5}$	74.3$_{\pm 0.9}$	92.5$_{\pm 2.1}$	87.2$_{\pm 1.4}$	80.7$_{\pm 1.0}$
	Latency	612.1	539.1	345.8	189.9	44.1	12.3	306.9	44.9	30.0	107.0	105.5	72.6	149.1
Qwen3-8B → Gemma-4-E4B-it														
Single	Acc.	50.0$_{\pm 5.8}$	40.0$_{\pm 5.8}$	82.5$_{\pm 4.3}$	80.0$_{\pm 1.4}$	90.7$_{\pm 1.0}$	91.6$_{\pm 1.4}$	51.5$_{\pm 2.3}$	94.6$_{\pm 0.9}$	98.1$_{\pm 0.5}$	74.3$_{\pm 0.9}$	84.6$_{\pm 2.5}$	76.7$_{\pm 1.7}$	67.4$_{\pm 1.2}$
Text (80)	Acc.	46.7$_{\pm 5.8}$	36.7$_{\pm 5.8}$	80.0$_{\pm 4.3}$	80.2$_{\pm 1.2}$	90.0$_{\pm 0.8}$	91.0$_{\pm 1.3}$	51.0$_{\pm 2.0}$	94.2$_{\pm 0.8}$	97.6$_{\pm 0.6}$	74.3$_{\pm 0.8}$	84.1$_{\pm 2.1}$	67.7$_{\pm 1.8}$	**68.7**$_{\pm 1.3}$
	Latency	269.4	219.0	212.9	80.6	28.9	24.1	96.4	29.3	19.6	82.5	103.4	37.8	89.3
Training-free	Acc.	46.7$_{\pm 5.8}$	43.3$_{\pm 5.8}$	80.0$_{\pm 5.0}$	86.8$_{\pm 1.4}$	82.4$_{\pm 0.7}$	88.6$_{\pm 1.5}$	**57.1**$_{\pm 1.7}$	90.5$_{\pm 0.9}$	96.7$_{\pm 0.3}$	72.8$_{\pm 0.9}$	73.6$_{\pm 1.4}$	67.7$_{\pm 1.0}$	39.6$_{\pm 1.7}$
	Latency	230.7	202.6	**196.9**	66.2	20.0	21.1	51.1	20.0	**16.2**	71.6	111.0	38.0	85.0
DIALECT	Acc.	**63.3**$_{\pm 3.3}$	**60.0**$_{\pm 3.3}$	**95.0**$_{\pm 4.3}$	**91.0**$_{\pm 1.0}$	**94.3**$_{\pm 0.5}$	**92.0**$_{\pm 0.8}$	56.9$_{\pm 1.5}$	**95.8**$_{\pm 0.6}$	**98.8**$_{\pm 0.4}$	**74.6**$_{\pm 0.5}$	**85.8**$_{\pm 1.8}$	**77.8**$_{\pm 1.0}$	67.9$_{\pm 0.9}$
	Latency	**209.4**	**198.3**	197.2	**64.3**	**18.6**	22.3	51.0	**15.4**	**16.2**	**63.7**	**89.2**	**36.3**	**81.5**
Full text	Acc.	60.0$_{\pm 3.3}$	56.7$_{\pm 5.8}$	92.5$_{\pm 4.3}$	94.8$_{\pm 0.8}$	93.6$_{\pm 0.9}$	92.8$_{\pm 1.3}$	57.4$_{\pm 2.0}$	96.3$_{\pm 0.8}$	99.8$_{\pm 0.3}$	75.6$_{\pm 0.9}$	85.8$_{\pm 1.9}$	78.9$_{\pm 1.8}$	69.5$_{\pm 1.4}$
	Latency	391.6	301.2	289.7	151.5	36.8	37.6	107.2	35.0	31.2	123.2	169.6	78.0	144.0
Qwen3-8B → Ministral-3B														
Single	Acc.	53.3$_{\pm 5.8}$	35.6$_{\pm 5.1}$	82.5$_{\pm 4.3}$	94.8$_{\pm 1.2}$	92.6$_{\pm 0.8}$	89.6$_{\pm 1.4}$	39.2$_{\pm 2.4}$	90.6$_{\pm 0.9}$	96.7$_{\pm 0.5}$	58.9$_{\pm 0.9}$	88.4$_{\pm 2.7}$	89.1$_{\pm 1.9}$	68.2$_{\pm 1.2}$
Text (80)	Acc.	43.3$_{\pm 5.8}$	36.7$_{\pm 3.3}$	85.0$_{\pm 4.3}$	94.0$_{\pm 1.4}$	92.2$_{\pm 0.9}$	89.0$_{\pm 1.4}$	38.4$_{\pm 1.8}$	90.2$_{\pm 0.8}$	96.3$_{\pm 0.5}$	58.5$_{\pm 1.0}$	87.8$_{\pm 2.4}$	89.5$_{\pm 1.9}$	68.2$_{\pm 1.6}$
	Latency	309.4	245.0	188.9	101.1	32.6	21.6	171.0	35.1	19.2	79.8	86.5	45.9	81.9
Training-free	Acc.	53.3$_{\pm 5.8}$	40.0$_{\pm 5.8}$	**90.0**$_{\pm 4.3}$	92.2$_{\pm 1.4}$	**93.4**$_{\pm 0.8}$	90.0$_{\pm 1.1}$	47.3$_{\pm 2.0}$	90.8$_{\pm 0.3}$	96.9$_{\pm 0.5}$	**60.2**$_{\pm 0.8}$	88.4$_{\pm 2.4}$	90.3$_{\pm 1.7}$	69.3$_{\pm 1.7}$
	Latency	236.7	230.6	**132.2**	**95.4**	21.8	18.0	133.7	29.9	18.7	64.5	77.7	40.0	77.0
DIALECT	Acc.	**56.7**$_{\pm 3.3}$	**43.3**$_{\pm 3.3}$	89.2$_{\pm 3.8}$	**95.4**$_{\pm 0.8}$	**93.4**$_{\pm 0.5}$	**90.6**$_{\pm 0.8}$	**48.0**$_{\pm 1.3}$	**91.0**$_{\pm 0.6}$	**97.1**$_{\pm 0.4}$	59.6$_{\pm 0.5}$	**89.0**$_{\pm 2.1}$	**90.7**$_{\pm 1.4}$	**70.1**$_{\pm 0.9}$
	Latency	**201.0**	**220.8**	**132.2**	96.1	**20.3**	**15.7**	**119.1**	**24.4**	**16.4**	**62.0**	**71.9**	**39.0**	**75.8**
Full text	Acc.	60.0$_{\pm 3.3}$	43.3$_{\pm 5.8}$	95.0$_{\pm 4.3}$	95.7$_{\pm 1.1}$	94.1$_{\pm 0.9}$	91.2$_{\pm 1.3}$	49.0$_{\pm 2.0}$	91.4$_{\pm 0.8}$	97.5$_{\pm 0.7}$	60.1$_{\pm 0.9}$	89.6$_{\pm 2.2}$	91.4$_{\pm 1.0}$	70.9$_{\pm 1.5}$
	Latency	497.9	502.4	305.5	193.9	37.5	28.8	222.9	44.9	30.2	116.8	136.4	76.7	144.8
Gemma-4-E4B-it → Ministral-3B														
Single	Acc.	53.3$_{\pm 5.8}$	35.6$_{\pm 5.1}$	82.5$_{\pm 4.3}$	94.8$_{\pm 1.2}$	92.6$_{\pm 0.8}$	89.6$_{\pm 1.4}$	39.2$_{\pm 2.4}$	90.6$_{\pm 0.9}$	96.7$_{\pm 0.5}$	58.9$_{\pm 0.9}$	88.4$_{\pm 2.7}$	89.1$_{\pm 1.9}$	68.2$_{\pm 1.2}$
Text (80)	Acc.	43.3$_{\pm 5.8}$	35.6$_{\pm 5.1}$	77.5$_{\pm 5.0}$	93.8$_{\pm 1.1}$	92.0$_{\pm 0.7}$	89.2$_{\pm 1.2}$	38.4$_{\pm 1.7}$	90.2$_{\pm 0.7}$	96.3$_{\pm 0.6}$	58.5$_{\pm 0.7}$	87.8$_{\pm 2.2}$	88.7$_{\pm 1.4}$	67.7$_{\pm 1.0}$
	Latency	289.0	262.0	229.8	85.5	26.7	18.4	146.2	35.8	19.5	83.0	111.1	45.7	72.2
Training-free	Acc.	33.3$_{\pm 5.8}$	36.7$_{\pm 3.3}$	65.0$_{\pm 4.3}$	71.6$_{\pm 1.5}$	69.9$_{\pm 0.8}$	32.4$_{\pm 1.2}$	28.1$_{\pm 1.2}$	90.0$_{\pm 0.6}$	96.3$_{\pm 0.4}$	58.9$_{\pm 0.4}$	43.9$_{\pm 2.7}$	53.3$_{\pm 2.0}$	39.6$_{\pm 1.6}$
	Latency	222.4	254.1	203.0	98.1	19.2	22.6	106.1	27.0	17.1	75.4	97.0	50.2	64.7
DIALECT	Acc.	**56.7**$_{\pm 3.3}$	**40.0**$_{\pm 3.3}$	**90.0**$_{\pm 4.3}$	**95.4**$_{\pm 0.9}$	**93.0**$_{\pm 0.5}$	**90.0**$_{\pm 0.8}$	**40.9**$_{\pm 1.5}$	**91.1**$_{\pm 0.6}$	**97.2**$_{\pm 0.4}$	**59.4**$_{\pm 0.5}$	**89.6**$_{\pm 1.6}$	**90.0**$_{\pm 1.1}$	**68.7**$_{\pm 1.1}$
	Latency	**200.3**	**203.5**	**177.2**	**70.4**	**14.4**	**10.9**	**97.5**	**24.8**	**16.7**	**65.0**	**91.3**	**28.7**	**49.0**
Full text	Acc.	60.0$_{\pm 5.8}$	43.3$_{\pm 5.8}$	92.5$_{\pm 4.3}$	96.0$_{\pm 1.4}$	93.2$_{\pm 1.0}$	92.8$_{\pm 1.5}$	42.4$_{\pm 2.2}$	91.6$_{\pm 0.7}$	97.8$_{\pm 0.6}$	59.9$_{\pm 0.8}$	91.5$_{\pm 2.2}$	90.9$_{\pm 1.6}$	69.5$_{\pm 1.5}$
	Latency	402.7	481.0	254.3	162.2	31.1	23.4	190.2	45.6	30.7	122.6	197.2	67.4	100.8

of 1.07×, 1.13×, 1.11×, and 1.31×, respectively (all $p < .001$). Paired accuracy and latency comparisons are significant for all four pairs against both baselines. Compatible vocabularies make the baselines competitive within the Qwen family. For cross-family pairs with lower vocabulary overlap (Table 3), show larger performance drops, particularly Gemma → Ministral and Qwen → Gemma, including results below single-agent performance. This pattern is consistent with the limitations of these methods, which discard contributions from unmatched tokens. Our learned bridge instead maps the source continuous representation without discarding these contributions (Table 4).

Against C2C, our method achieves higher mean accuracy on all 13 benchmarks and lower overall latency, with mean accuracy gains of 7.99, 9.15, 5.85, and 10.60 percentage points (all $p < .001$) and geometric-mean speedups of 1.03×, 1.07×, 1.08×, and 1.27×, respectively (all $p < .001$). These gains show that effective cross-model communication can translate continuous messages into the receiver's input space without exchanging intermediate hidden states or KV caches (Table 4).

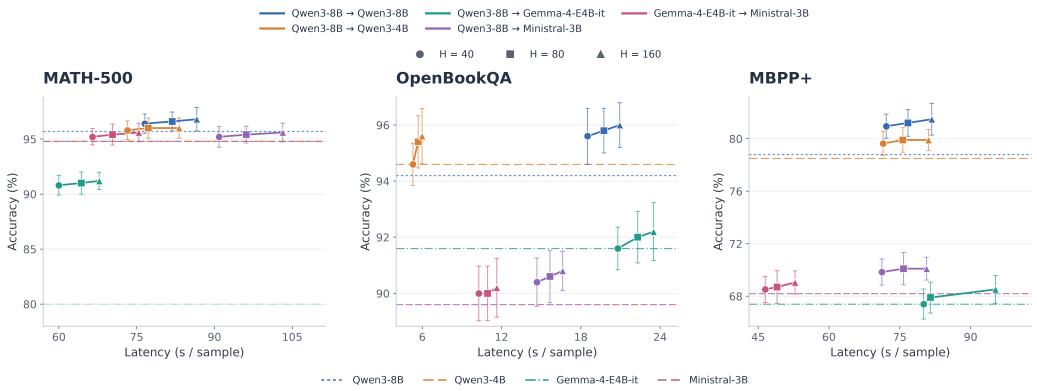


Figure 2: Accuracy–latency trade-offs for DIALECT across five pairs with communication budgets $H \in \{40, 80, 160\}$ (M $\pm$ SD, 3 seeds). Horizontal lines indicate single-agent target accuracy.

Compared with full-text communication, DIALECT achieves lower accuracy but reduces latency across all settings. The mean accuracy drops of 0.71 percentage points (99.16% retention), 0.03 percentage points (99.96% retention), 1.17 percentage points (98.52% retention), and 1.49 percentage points (98.10% retention), respectively. Wilcoxon-based TOST supports equivalence within ± 2 percentage points for the center of the paired accuracy differences in all four cross-model pairs ($p < .001$ for the three pairs with Qwen3-8B as the source and $p = .0018$ for Gemma $\rightarrow$ Ministral). The geometric-mean speedups are $2.13\times$, $1.90\times$, $1.99\times$, and $2.02\times$, respectively (all $p < .001$). These results show that short continuous thoughts can retain much of the benefit of communication between different models at lower latency.

5.3 Accuracy–Latency Trade-offs Across Token Budgets

Figure 2 shows the accuracy–latency trade-off for all pairs on one benchmark per category; remaining benchmarks appear in Figure 4. Increasing the communication budget improves or maintains mean accuracy across all pairs and categories while increasing latency in every setting, enabling an accuracy–latency balance. Mixed-effects analysis of individual seed-level results confirms a positive accuracy–latency association ($p < .001$). Adjacent-budget comparisons appear in Section T.

6 Conclusion

Inspired by nonlinguistic communication in the brain, we introduced DIALECT, a framework for continuous-space communication between agents built from the same or different LLMs. DIALECT combines stochastic soft-token generation with a learned cross-model bridge, keeping both LLMs frozen and requiring no training in the same-model setting. Across 13 benchmarks covering mathematical reasoning, knowledge question answering, and code generation, DIALECT improves accuracy over the single-agent target by a median of 2.69 percentage points across model pairs. Compared with budget-matched text transfer, it achieves higher accuracy while running 1.28–$1.41\times$ faster. Against full-text communication, it retains over 98% of benchmark-averaged accuracy across all pairs while achieving a median speedup of $2.02\times$, with paired equivalence tests supporting statistically equivalent accuracy in all model pairs. Training-free and CIPHER-Cross remain competitive within the Qwen family but show larger performance drops for pairs with lower vocabulary overlap, where DIALECT achieves mean accuracy gains up to 21.77 and 23.50 percentage points, respectively. Our learned bridge instead maps the source continuous representation into the target's input space without discarding contributions from unmatched tokens. The comparison with C2C further supports embedding-based thought transfer as an effective alternative to exchanging internal KV caches. Lastly, varying the communication budget traces an accuracy–latency frontier, letting a deployment trade inference time for accuracy.

AI USE STATEMENT

In this work, we used generative AI tools to assist with drafting portions of the manuscript and to improve grammar, clarity, and readability. We used GPT-5.2 as an automated answer judge for mathematical reasoning and knowledge tasks (Section 4.4 and Section G). We also used generative AI tools to assist with coding and experimental implementation. We reviewed the AI-assisted code and manuscript revisions. We take responsibility for the final content of this work, including text, claims, and artifacts produced with the aid of generative AI.

REPRODUCIBILITY STATEMENT

We provide our code in the supplementary material to support reproduction of our experiments. Section 3 and Sections A to D describe the continuous generation procedure, bridge architecture, and training objectives. Section 4 and Sections E to N document the experimental setup, benchmarks, prompts, baseline adaptations, and timing protocol.

REFERENCES

Jacob Austin, Augustus Odena, Maxwell Nye, Maarten Bosma, Henryk Michalewski, David Dohan, Ellen Jiang, Carrie Cai, Michael Terry, Quoc Le, et al. Program synthesis with large language models. *arXiv preprint arXiv:2108.07732*, 2021.

Yamini Bansal, Preetum Nakkiran, and Boaz Barak. Revisiting model stitching to compare neural representations. *Advances in neural information processing systems*, 34:225–236, 2021.

Yael Benn, Anna A Ivanova, Oliver Clark, Zachary Mineroff, Chloe Seikus, Jack Santos Silva, Rosemary Varley, and Evelina Fedorenko. The language network is not engaged in object categorization. *Cerebral Cortex*, 33(19):10380–10400, 2023.

Akhiad Bercovich, Itay Levy, Izik Golan, Mohammad Dabbah, Ran El-Yaniv, Omri Puny, Ido Galil, Zach Moshe, Tomer Ronen, Najeeb Nabwani, Ido Shahaf, Oren Tropp, Ehud Karpas, Ran Zilberstein, Jiaqi Zeng, Soumye Singhal, Alexander Bukharin, Yian Zhang, Tugrul Konuk, Gerald Shen, Ameya Sunil Mahabaleshwarkar, Bilal Kartal, Yoshi Suhara, Olivier Delalleau, Zijia Chen, Zhilin Wang, David Mosallanezhad, Adi Renduchintala, Haifeng Qian, Dima Rekesh, Fei Jia, Somshubra Majumdar, Vahid Noroozi, Wasi Uddin Ahmad, Sean Narenthiran, Aleksander Ficek, Mehrzad Samadi, Jocelyn Huang, Siddhartha Jain, Igor Gitman, Ivan Moshkov, Wei Du, Shubham Toshniwal, George Armstrong, Branislav Kisacanin, Matvei Novikov, Daria Gitman, Evelina Bakhturina, Jane Polak Scowcroft, John Kamalu, Dan Su, Kezhi Kong, Markus Kliegl, Rabeeh Karimi, Ying Lin, Sanjeev Satheesh, Jupinder Parmar, Pritam Gundecha, Brandon Norick, Joseph Jennings, Shrimai Prabhumoye, Syeda Nahida Akter, Mostofa Patwary, Abhinav Khattar, Deepak Narayanan, Roger Waleffe, Jimmy Zhang, Bor-Yiing Su, Guyue Huang, Terry Kong, Parth Chadha, Sahil Jain, Christine Harvey, Elad Segal, Jining Huang, Sergey Kashirsky, Robert McQueen, Izzy Putterman, George Lam, Arun Venkatesan, Sherry Wu, Vinh Nguyen, Manoj Kilaru, Andrew Wang, Anna Warno, Abhilash Somasamudramath, Sandip Bhaskar, Maka Dong, Nave Assaf, Shahar Mor, Omer Ullman Argov, Scot Junkin, Oleksandr Romanenko, Pedro Larroy, Monika Katariya, Marco Rovinelli, Viji Balas, Nicholas Edelman, Anahita Bhiwandiwalla, Muthu Subramaniam, Smita Ithape, Karthik Ramamoorthy, Yuting Wu, Suguna Varshini Velury, Omri Almog, Joyjit Daw, Denys Fridman, Erick Galinkin, Michael Evans, Katherine Luna, Leon Derczynski, Nikki Pope, Eileen Long, Seth Schneider, Guillermo Siman, Tomasz Grzegorzek, Pablo Ribalta, Monika Katariya, Joey Conway, Trisha Saar, Ann Guan, Krzysztof Pawelec, Shyamala Prayaga, Oleksii Kuchaiev, Boris Ginsburg, Oluwatobi Olabiyi, Kari Briski, Jonathan Cohen, Bryan Catanzaro, Jonah Alben, Yonatan Geifman, Eric Chung, and Chris Alexiuk. Llama-nemotron: Efficient reasoning models, 2025. URL https://arxiv.org/abs/2505.00949.

Alan Chen, Jack Merullo, Alessandro Stolfo, and Ellie Pavlick. Transferring linear features across language models with model stitching. *Advances in Neural Information Processing Systems*, 38: 48531–48563, 2026a.

Justin Chen, Swarnadeep Saha, and Mohit Bansal. Reconcile: Round-table conference improves reasoning via consensus among diverse llms. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 7066–7085, 2024.

Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde De Oliveira Pinto, Jared Kaplan, Harri Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman, et al. Evaluating large language models trained on code. *arXiv preprint arXiv:2107.03374*, 2021.

Siyi Chen, Xiaoyan Zhang, Meng Wu, Jonathan Tremblay, Valts Blukis, Stan Birchfield, Rene Vidal, Alvaro Velasquez, Sijia Liu, and Qing Qu. See what i see, know what i think: Dense latent communication across heterogeneous agents. *arXiv preprint arXiv:2606.13594*, 2026b.

Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Oyvind Tafjord. Think you have solved question answering? try arc, the ai2 reasoning challenge. *arXiv preprint arXiv:1803.05457*, 2018.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*, 2021.

Lucio M Dery, Zohar Yahav, Henry Prior, Qixuan Feng, Jiajun Shen, and Arthur Szlam. Latent space communication via kv cache alignment. *arXiv preprint arXiv:2601.06123*, 2026.

Yilun Du, Shuang Li, Antonio Torralba, Joshua B Tenenbaum, and Igor Mordatch. Improving factuality and reasoning in language models through multiagent debate. *arXiv preprint arXiv:2305.14325*, 2023.

Zhuoyun Du, Runze Wang, Huiyu Bai, Zouying Cao, Xiaoyong Zhu, Yu Cheng, Bo Zheng, Wei Chen, and Haochao Ying. Enabling agents to communicate entirely in latent space. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 27106–27129, 2026.

Evelina Fedorenko and Rosemary Varley. Language and thought are not the same thing: evidence from neuroimaging and neurological patients. *Annals of the New York Academy of Sciences*, 1369 (1):132–153, 2016.

Evelina Fedorenko, Steven T Piantadosi, and Edward AF Gibson. Language is primarily a tool for communication rather than thought. *Nature*, 630(8017):575–586, 2024.

Tianyu Fu, Zihan Min, Hanling Zhang, Jichao Yan, Guohao Dai, Wanli Ouyang, and Yu Wang. Cache-to-cache: Direct semantic communication between large language models. In *International Conference on Learning Representations*, volume 2026, pp. 43130–43158, 2026.

Shibo Hao, Sainbayar Sukhbaatar, DiJia Su, Xian Li, Zhiting Hu, Jason Weston, and Yuandong Tian. Training large language models to reason in a continuous latent space. *arXiv preprint arXiv:2412.06769*, 2024.

Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. Measuring mathematical problem solving with the math dataset. *arXiv preprint arXiv:2103.03874*, 2021.

Sirui Hong, Mingchen Zhuge, Jonathan Chen, Xiawu Zheng, Yuheng Cheng, Jinlin Wang, Ceyao Zhang, Steven Yau, Zijuan Lin, Liyang Zhou, et al. Metagpt: Meta programming for a multi-agent collaborative framework. In *International Conference on Learning Representations*, volume 2024, pp. 23247–23275, 2024.

Hugging Face H4. AIME 2024, 2024. URL `https://huggingface.co/datasets/HuggingFaceH4/aime_2024`.

Di Jin, Eileen Pan, Nassim Oufattole, Wei-Hung Weng, Hanyi Fang, and Peter Szolovits. What disease does this patient have? a large-scale open domain question answering dataset from medical exams. *arXiv preprint arXiv:2009.13081*, 2020.

Su-Hyeon Kim, Jiwan Mun, and Yo-Sub Han. One adapter pair per model: A universal activation interface for language models. *arXiv preprint arXiv:2608.09521*, 2026.

Karel Lenc and Andrea Vedaldi. Understanding image representations by measuring their equivariance and equivalence. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 991–999, 2015.

Guohao Li, Hasan Hammoud, Hani Itani, Dmitrii Khizbullin, and Bernard Ghanem. Camel: Communicative agents for" mind" exploration of large language model society. *Advances in neural information processing systems*, 36:51991–52008, 2023.

Xiang Lisa Li and Percy Liang. Prefix-tuning: Optimizing continuous prompts for generation. In *Proceedings of the 59th annual meeting of the association for computational linguistics and the 11th international joint conference on natural language processing (volume 1: Long papers)*, pp. 4582–4597, 2021.

Tian Liang, Zhiwei He, Wenxiang Jiao, Xing Wang, Yan Wang, Rui Wang, Yujiu Yang, Shuming Shi, and Zhaopeng Tu. Encouraging divergent thinking in large language models through multi-agent debate. In *Proceedings of the 2024 conference on empirical methods in natural language processing*, pp. 17889–17904, 2024.

Hunter Lightman, Vineet Kosaraju, Yuri Burda, Harrison Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. Let's verify step by step. In *International Conference on Learning Representations*, volume 2024, pp. 39578–39601, 2024.

Alexander H Liu, Kartik Khandelwal, Sandeep Subramanian, Victor Jouault, Abhinav Rastogi, Adrien Sadé, Alan Jeffares, Albert Jiang, Alexandre Cahill, Alexandre Gavaudan, et al. Ministral 3. *arXiv preprint arXiv:2601.08584*, 2026a.

Jiawei Liu, Chunqiu Steven Xia, Yuyao Wang, and Lingming Zhang. Is your code generated by chatgpt really correct? rigorous evaluation of large language models for code generation. *Advances in neural information processing systems*, 36:21558–21572, 2023.

Jiyao Liu, Qi Zhang, Yaoyi Jia, Ziwen Kan, and Song Wang. Dual-cache latent space communication between heterogeneous language models. *arXiv preprint arXiv:2608.20617*, 2026b.

Valentino Maiorca, Luca Moschella, Antonio Norelli, Marco Fumero, Francesco Locatello, and Emanuele Rodolà. Latent space translation via semantic alignment. *Advances in Neural Information Processing Systems*, 36:55394–55414, 2023.

Math-AI. AMC23, 2023. URL `https://huggingface.co/datasets/math-ai/amc23`.

Todor Mihaylov, Peter Clark, Tushar Khot, and Ashish Sabharwal. Can a suit of armor conduct electricity? a new dataset for open book question answering. In *EMNLP*, 2018.

Luca Moschella, Valentino Maiorca, Marco Fumero, Antonio Norelli, Francesco Locatello, and Emanuele Rodolà. Relative representations enable zero-shot latent space communication. *arXiv preprint arXiv:2209.15430*, 2022.

Yanwen Peng, Delvin Ce Zhang, Xi Wang, and Nikolaos Aletras. Statebridge: Training-free hidden-state alignment for latent communication in llm multi-agent systems. *arXiv preprint arXiv:2608.13317*, 2026.

Chau Pham, Boyi Liu, Yingxiang Yang, Zhengyu Chen, Tianyi Liu, Jianbo Yuan, Bryan A. Plummer, Zhaoran Wang, and Hongxia Yang. Let models speak ciphers: Multiagent debate through embeddings. In *International Conference on Learning Representations (ICLR)*, 2024.

Chen Qian, Wei Liu, Hongzhang Liu, Nuo Chen, Yufan Dang, Jiahao Li, Cheng Yang, Weize Chen, Yusheng Su, Xin Cong, et al. Chatdev: Communicative agents for software development. In *Proceedings of the 62nd annual meeting of the association for computational linguistics (volume 1: Long papers)*, pp. 15174–15186, 2024.

Vignav Ramesh and Kenneth Li. Communicating activations between language model agents. *arXiv preprint arXiv:2501.14082*, 2025.

David Rein, Betty Li Hou, Asa Cooper Stickland, Jackson Petty, Richard Yuanzhe Pang, Julien Dirani, Julian Michael, and Samuel R Bowman. Gpqa: A graduate-level google-proof q&a benchmark. *arXiv preprint arXiv:2311.12022*, 2023.

João D Semedo, Amin Zandvakili, Christian K Machens, Byron M Yu, and Adam Kohn. Cortical areas interact through a communication subspace. *Neuron*, 102(1):249–259, 2019.

Zhenyi Shen, Hanqi Yan, Linhai Zhang, Zhanghao Hu, Yali Du, and Yulan He. Codi: Compressing chain-of-thought into continuous space via self-distillation. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pp. 677–693, 2025.

Gemma Team, Sherif El Abd, Vaibhav Aggarwal, Robin Algayres, Alek Andreev, Olivier Bachem, Ian Ballantyne, Cormac Brick, Victor Cărbune, Michelle Casbon, et al. Gemma 4 technical report. *arXiv preprint arXiv:2607.02770*, 2026.

Junlin Wang, Jue Wang, Ben Athiwaratkun, Ce Zhang, and James Y Zou. Mixture-of-agents enhances large language model capabilities. In *International Conference on Learning Representations*, volume 2025, pp. 33944–33963, 2025.

Lei Wang, Wanyu Xu, Yihuai Lan, Zhiqiang Hu, Yunshi Lan, Roy Ka-Wei Lee, and Ee-Peng Lim. Plan-and-solve prompting: Improving zero-shot chain-of-thought reasoning by large language models. In *Proceedings of the 61st annual meeting of the association for computational linguistics (volume 1: Long papers)*, pp. 2609–2634, 2023.

Junhong Wu, Jinliang Lu, Zixuan Ren, Gangqiang Hu, Zhi Wu, Dai Dai, and hua wu. Llms are single-threaded reasoners: Demystifying the working mechanism of soft thinking. In C. Vondrick, B. Hariharan, C. Raffel, L. Pinto, D. Yang, and A. Faust (eds.), *International Conference on Learning Representations*, volume 2026, pp. 4533–4548, 2026. URL https://proceedings.iclr.cc/paper_files/paper/2026/file/0895de150660b6c2aef9b0f098cb1dcd-Paper-Conference.pdf.

Qingyun Wu, Gagan Bansal, Jieyu Zhang, Yiran Wu, Beibin Li, Erkang Zhu, Li Jiang, Xiaoyun Zhang, Shaokun Zhang, Jiale Liu, et al. Autogen: Enabling next-gen llm applications via multi-agent conversation. *arXiv preprint arXiv:2308.08155*, 2023.

Yige Xu, Xu Guo, Zhiwei Zeng, and Chunyan Miao. Softcot: Soft chain-of-thought for efficient reasoning with llms. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 23336–23351, 2025a.

Yige Xu, Xu Guo, Zhiwei Zeng, and Chunyan Miao. Softcot++: Test-time scaling with soft chain-of-thought reasoning. *arXiv preprint arXiv:2505.11484*, 2025b.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, et al. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*, 2025.

Wooseong Yang, Wei-Chieh Huang, Weizhi Zhang, Yu Wang, Philip S Yu, and Junhyun Lee. Xbridge: Entity-grounded latent bridge for heterogeneous llm communication. *arXiv preprint arXiv:2608.11676*, 2026.

Sami R Yousif and Russell A Epstein. Cognitive maps as a medium for thought. *Trends in Cognitive Sciences*, 2026.

Yifan Zhang and Math-AI Team. American invitational mathematics examination (AIME) 2025, 2025. URL https://huggingface.co/datasets/math-ai/aime25.

Zhen Zhang, Xuehai He, Weixiang Yan, Ao Shen, Chenyang Zhao, and Xin Wang. Soft thinking: Unlocking the reasoning potential of llms in continuous concept space. *Advances in Neural Information Processing Systems*, 38:168990–169012, 2026.

Yujia Zheng, Zhuokai Zhao, Zijian Li, Yaqi Xie, Mingze Gao, Lizhu Zhang, and Kun Zhang. Thought communication in multiagent collaboration. *Advances in Neural Information Processing Systems*, 38:123389–123418, 2026.

Yufan Zhuang, Liyuan Liu, Chandan Singh, Jingbo Shang, and Jianfeng Gao. Mixture of inputs: Text generation beyond discrete token sampling. *Advances in Neural Information Processing Systems*, 38:26363–26387, 2026.

Jiaru Zou, Xiyuan Yang, Ruizhong Qiu, Gaotang Li, Katherine Tieu, Pan Lu, Ke Shen, Hanghang Tong, Yejin Choi, Jingrui He, et al. Latent collaboration in multi-agent systems. *arXiv preprint arXiv:2511.20639*, 2025.

Jiaru Zou, Rui Pan, Ruizhong Qiu, Pan Lu, Shizhe Diao, Jindong Jiang, Hanghang Tong, Tong Zhang, Markus J Buehler, Jingrui He, et al. Recursive multi-agent systems. *arXiv preprint arXiv:2604.25917*, 2026.

A Stochastic Soft-Token Generation

Following Wu et al. (2026), we use Dirichlet and Gumbel–Softmax sampling to construct embedding weights from the truncated and renormalized distribution $Q^t = (q_i^t)_{i\in\mathcal{S}^t}$ defined in Section 3.2. We use scaled Dirichlet mixtures for same-model communication and Gumbel–Softmax mixtures for cross-model communication.

Scaled Dirichlet sampling. We sample embedding weights

$$W^t = (w_i^t)_{i\in\mathcal{S}^t} \sim \mathrm{Dirichlet}(\alpha Q^t), \tag{5}$$

where $\alpha > 0$ controls their concentration around the truncated token distribution Q^t. Since weighted averaging can reduce embedding norms, we rescale the resulting mixture to match the mean norm of the model's input embeddings:

$$\tilde{e}_D^t = \frac{\bar{r}_E}{\left\|\sum_{i\in\mathcal{S}^t} w_i^t e_i\right\|_2} \sum_{i\in\mathcal{S}^t} w_i^t e_i, \qquad \bar{r}_E = \frac{1}{|\mathcal{V}|}\sum_{j\in\mathcal{V}} \|e_j\|_2. \tag{6}$$

Thus, the rescaling factor in Section 3.2 is $c^t = \bar{r}_E / \left\|\sum_{i\in\mathcal{S}^t} w_i^t e_i\right\|_2$. This rescaling preserves the direction of the sampled mixture while setting its norm to $\bar{r}_E$.

Gumbel–Softmax sampling. We independently sample $g_i^t \sim \mathrm{Gumbel}(0,1)$ for each retained token $i \in \mathcal{S}^t$ and construct embedding weights using

$$w_i^t = \frac{\exp((\log q_i^t + g_i^t)/\tau)}{\sum_{j\in\mathcal{S}^t}\exp((\log q_j^t + g_j^t)/\tau)}, \tag{7}$$

where $\tau > 0$ controls the sharpness of the sampled weights. We use the resulting weighted combination of input embeddings without additional norm rescaling:

$$\tilde{e}_G^t = \sum_{i\in\mathcal{S}^t} w_i^t e_i. \tag{8}$$

Thus, $c^t = 1$ for this variant. Sampling hyperparameters are reported in Section 4.5

B Bridge Architecture

Input standardization. We standardize source representations coordinate-wise:

$$\bar{u}_j = \frac{u_j - \mu_j}{s_j}, \qquad \mu_j = \frac{1}{N}\sum_{n=1}^{N} u_{n,j}, \qquad s_j = \sqrt{\frac{1}{N}\sum_{n=1}^{N}(u_{n,j}-\mu_j)^2 + 10^{-8}}. \tag{9}$$

The statistics are computed over the retained source vectors in the stored pretraining representation corpus, before its training/validation/test partitioning. They are stored as non-trainable checkpoint buffers and remain fixed during subsequent training and inference.

Main embedding branch. The bridge maps each source soft token u to the target input-embedding space by predicting its direction and magnitude separately. Let $\bar{u}$ denote the source vector after the bridge's input standardization. The direction branch is

$$r_\theta(\bar{u}) = W_0\bar{u} + b_0 + W_2\,\mathrm{Dropout}[\mathrm{GELU}(W_1\,\mathrm{RMSNorm}(\bar{u}) + b_1)] + b_2. \tag{10}$$

The residual branch uses non-affine RMS normalization with $\epsilon = 10^{-6}$, a hidden width of 4096, GELU activation, and dropout with probability 0.1. Its final projection is initialized to zero. The magnitude network g_θ consists of layer normalization, a 256-unit linear layer, GELU activation, and a scalar output layer. It defines

$$a_\theta^{\mathrm{raw}}(\bar{u}) = a_{\min} + \mathrm{softplus}(g_\theta(\bar{u})), \qquad a_\theta(\bar{u}) = \min\{a_{\max}, a_\theta^{\mathrm{raw}}(\bar{u})\}, \tag{11}$$

where $a_{\min} = 10^{-4}$. The mapped main embedding is

$$F_\theta^{\mathrm{main}}(u) = a_\theta(\bar{u})\,\frac{r_\theta(\bar{u})}{\max(\|r_\theta(\bar{u})\|_2, 10^{-6})}. \tag{12}$$

The scalar output weights are initialized to zero, and the bias sets the initial magnitude. Initial magnitudes and upper bounds are

Target	Initial magnitude	$a_{\max}$
Qwen3-4B	1.0973718	2.5
Ministral	0.385707	1.0
Gemma	51.2	100.0

During training, a straight-through operation applies the cap in the forward pass while passing gradients through the uncapped magnitude. The cap itself enforces the bound on the mapped main embedding.

Input and output dimensions. The main branch maps 4096 source coordinates to 2560 target coordinates for Qwen3-8B-to-Qwen3-4B, 4096 to 3072 for Qwen3-8B-to-Ministral, 2560 to 3072 for Gemma-to-Ministral, and 4096 to 2560 for Qwen3-8B-to-Gemma.

Per-layer output head. For Gemma targets, an additional head predicts residuals for the native per-layer embeddings (PLE). It receives the standardized source vector and uses non-affine RMS normalization with $\epsilon = 10^{-6}$, a 4096-unit linear layer, GELU activation, dropout with probability 0.1, and a final linear projection initialized to zero. It produces $42 \times 256 = 10752$ coordinates, reshaped into a 256-dimensional residual for each of 42 layers. Together with the 2560-dimensional main embedding, the bridge outputs 13312 coordinates per communication position. The PLE residuals do not use a separate magnitude predictor or the main branch's norm cap.

Let $P_B[\mathrm{PAD}, j]$ be the target padding token's native PLE at layer j, and let $\Delta P_\theta(u)_j$ be the predicted residual. At communication position t, the target receives

$$p_B^{t,j} = P_B[\mathrm{PAD}, j] + \Delta P_\theta(\tilde{e}_A^t)_j. \tag{13}$$

At initialization, the head therefore supplies the padding token's native PLE. Prompt and response positions retain the PLE associated with their token IDs. The target's context-dependent per-layer projection remains unchanged and frozen.

Source per-layer inputs. When Gemma is the source, continuous generation requires both main embeddings and per-layer inputs. We obtain both from the model's native embedding functions and combine them with the same sampled token weights. In particular, $p_A^{t,j} = \sum_{k \in \mathcal{V}_A} w_{A,k}^t P_A[k, j]$, with zero weights outside the retained token set. Native embedding scaling is preserved. Both embedding types are fed back into the source; only the main soft token is passed to the bridge.

C Representation Pair Construction

Standalone pretraining uses stored pairs $\mathcal{D}_{\mathrm{pair}} = \{(u_n, v_n)\}_{n=1}^N$. Each pair contains a source embedding u_n and a target reference v_n. The reference specifies the embedding that the bridge is trained to predict.

Shared-token pairs. For Qwen3-8B-to-Qwen3-4B, we run source continuous generation and apply each sampled token-weight vector to both models' input-embedding tables. The source mixture is used without additional norm rescaling; the target reference is rescaled to norm 1.0973718. This produces paired vectors from the same sampled token weights.

Sentence-boundary pairs. For model pairs with different tokenizers, we construct pretraining pairs from corresponding sentence boundaries in shared reference responses. We format each response using the respective model's chat template and identify sentence boundaries by splitting the raw response at whitespace following a period, question mark, or exclamation mark. Tokenizing each formatted prefix gives the corresponding boundary token for each model, accounting for tokenizer-added prefix tokens and batch padding. For each matched sentence boundary, we select the input embedding one token before the computed sentence-final token in each model's tokenization. For Gemma targets, the reference also includes native PLE residuals relative to the padding token's PLE.

Training data. We use the math, science, and code splits of the Llama-Nemotron-Post-Training-Dataset (Bercovich et al., 2025). Each domain contains 16,000 retained examples: 14,400 for end-to-end training, 320 for validation, and 1,280 held out. Splits use seed 42 and are shared across model pairs and communication lengths. Pretraining pairs are constructed only from the end-to-end training partition.

Pretraining objectives. All four model pairs receive representation-based pretraining using the alignment loss in eq. (3). We minimize $\mathcal{L}_{\text{pre}} = N^{-1} \sum_{n=1}^{N} \mathcal{A}(F_\theta(u_n), v_n)$, with $\beta_n = 0.1$ and $\beta_r = 0.25$. For Gemma targets, supervision includes both the main embeddings and per-layer residuals. This pretraining objective is separate from the additional online alignment term used during end-to-end training.

Pretraining optimization. We pretrain each bridge for up to 100 epochs using AdamW with a learning rate of 3×10^{-5}, weight decay of 0.01, and a batch size of 8,192 representation pairs. The stored representation pairs are randomly partitioned into training, validation, and test subsets with proportions $90\%/5\%/5\%$, using seed 42. Validation is performed after every epoch, and the checkpoint with the lowest validation loss is retained. We use `ReduceLROnPlateau` with a reduction factor of 0.5, patience of four validation checks, and a minimum learning rate of 10^{-6}. Training stops after eight consecutive validation checks without improvement.

D BRIDGE TRAINING DETAILS

After pretraining, we optimize the bridge within the source–target pipeline. For each example (x_i, y_i), the source generates a continuous thought of length H. The bridge maps its vectors to the target input space and inserts them after the target prompt. The target predicts the reference response, including its reasoning and final answer, using teacher forcing. Both language models remain frozen; only bridge parameters are updated.

Response supervision and message comparisons. Let L_i be the number of retained reference-response tokens. For condition s, define the per-example loss

$$C_i^s = -\frac{1}{L_i} \sum_{\ell=1}^{L_i} \log p_B(y_{i,\ell} \mid x_i, m_i^s, y_{i,<\ell}), \qquad s \in \{+, \text{shuf}, 0, B\}, \tag{14}$$

where m_i^s denotes the communication input under that condition. The matched condition $(+)$ uses the thought generated for x_i. The shuffled condition cyclically shifts mapped messages by one example within the physical batch. For Gemma targets, the main embeddings and PLE residuals are shifted together. The zero condition retains the communication positions but sets mapped main embeddings and predicted PLE residuals to zero; Gemma therefore retains the padding token's native PLE at those positions. The target-only condition (B) removes the communication positions. The question and supervised reference response are fixed across conditions.

We use the smooth margin penalty

$$R_m(a, b) = \tau_r \log\left(1 + \exp\left(\frac{m + a - b}{\tau_r}\right)\right), \qquad \tau_r = 0.1. \tag{15}$$

For a batch of B_{tr} examples, the component losses are

$$\begin{aligned}
\mathcal{L}_{\text{ans}} &= \frac{1}{B_{\text{tr}}} \sum_i C_i^+, \\
\mathcal{L}_{\text{shuf}} &= \frac{1}{B_{\text{tr}}} \sum_i R_{0.10}(C_i^+, C_i^{\text{shuf}}), \\
\mathcal{L}_0 &= \frac{1}{B_{\text{tr}}} \sum_i R_{0.05}(C_i^+, C_i^0), \\
\mathcal{L}_{\text{base}} &= \frac{1}{B_{\text{tr}}} \sum_i \max(0, C_i^+ - C_i^B).
\end{aligned} \tag{16}$$

The shuffled and zero comparisons encourage lower response loss with the matched message by the specified margins. The target-only comparison penalizes increased loss relative to omitting the message.

Text-plan teacher generation and selection. For each example, the source independently samples two text plans using temperature 0.6, top-p 0.95, and top-k 20. Each plan is decoded and retokenized for the target, which receives it as additional user context. Thus, the teacher distribution is produced by the target conditioned on a source-generated text plan, and the student distribution is produced by the same target conditioned on the mapped continuous thought.

Both conditions use teacher forcing on the full retained reference response. Let $C^T_{i,c}$ be the mean reference-response cross-entropy under candidate plan $c \in \{1, 2\}$. We select

$$c^*_i = \arg\min_{c\in\{1,2\}} C^T_{i,c}, \qquad \gamma_i = \mathbf{1}[C^T_{i,c^*_i} < C^+_i]. \tag{17}$$

Only examples for which the selected text-plan teacher has lower cross-entropy contribute to distillation. Teacher selection and the gate are excluded from gradient computation.

Distillation loss. We divide teacher and student logits by temperature $T = 2$ before applying softmax. At each reference-response position, we retain the selected teacher's $K = 32$ highest-probability token IDs and aggregate the remaining probability mass into one additional category. Student probabilities are grouped using the same token IDs, giving distributions $\hat{p}^T_{i,\ell}$ and $\hat{p}^+_{i,\ell}$ with $K + 1$ entries. We compute

$$\mathcal{L}_{\mathrm{KD}} = \frac{T^2}{\max(1, \sum_i \gamma_i)} \sum_i \frac{\gamma_i}{L_i} \sum_{\ell=1}^{L_i} D_{\mathrm{KL}}\left(\hat{p}^T_{i,\ell} \,\middle\|\, \hat{p}^+_{i,\ell}\right). \tag{18}$$

The loss is averaged over response positions within each selected example and then over selected examples. It is zero when no example passes the gate. Teacher and student positions correspond to the same reference tokens, even though their conditioning prefixes have different lengths.

Online embedding alignment. End-to-end alignment is enabled only for Qwen3-8B-to-Qwen3-4B. For each continuous generation step, we apply the source's sampled token weights to the target embedding table and rescale the resulting mixture to norm 1.0973718, obtaining $v_{i,t}$. These references are constructed from the current rollout, rather than sampled from the stored pretraining pairs. The alignment loss is

$$\mathcal{L}_{\mathrm{align}} = \frac{1}{B_{\mathrm{tr}} H} \sum_i \sum_{t=1}^{H} \mathcal{A}\big(F_\theta(\tilde{e}^t_{A,i}), v_{i,t}\big), \tag{19}$$

using eq. (3) with $\beta_n = 0.1$ and $\beta_r = 0.25$. Norms in the magnitude and relative-error terms are floored at 10^{-8}. For the other three model pairs, the online alignment term is disabled after representation-based pretraining. Their bridges are optimized through the prediction and communication losses, subject to the architectural norm constraint. For Gemma targets, this includes updating both the main embedding branch and the PLE head without an additional online PLE alignment loss.

Norm constraint and auxiliary penalty. The magnitude cap in section B bounds the mapped main embeddings. The implemented objective also includes

$$\mathcal{L}_{\mathrm{upper}} = \frac{1}{B_{\mathrm{tr}} H} \sum_i \sum_{t=1}^{H} \left[\max\left(0, \log \frac{\max(\|\tilde{e}^t_{B,i}\|_2, 10^{-8})}{a_{\max}}\right)\right]^2. \tag{20}$$

This penalty is evaluated on the capped main embeddings and uses the same bound as the architecture. It is therefore zero in exact arithmetic under the configured parameterization, apart from numerical deviations in implementation. We retain it here to specify the implemented objective; the architectural cap is the effective norm constraint.

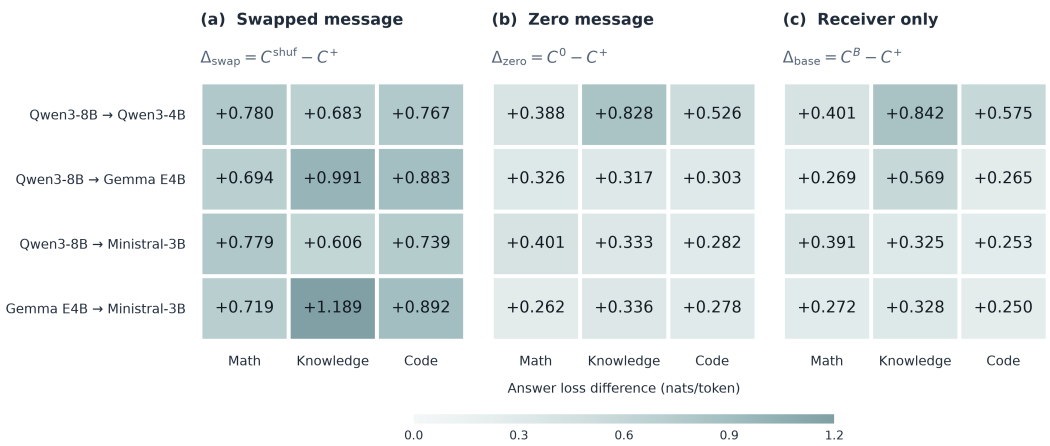


Figure 3: Validation diagnostics of message specificity at $H = 80$.

Complete objective and coefficients. The objective is

$$\begin{aligned} \mathcal{L} &= \mathcal{L}_{\text{pred}} + \mathcal{L}_{\text{comm}} + \mathcal{L}_{\text{repr}}, \\ \mathcal{L}_{\text{pred}} &= \mathcal{L}_{\text{ans}} + 0.1\mathcal{L}_{\text{KD}}, \\ \mathcal{L}_{\text{comm}} &= \mathcal{L}_{\text{shuf}} + 0.5\mathcal{L}_{0} + 2\mathcal{L}_{\text{base}}, \\ \mathcal{L}_{\text{repr}} &= 0.25\mathcal{L}_{\text{align}} + 0.05\mathcal{L}_{\text{upper}}, \end{aligned} \tag{21}$$

Parameter updates and optimization. Gradients pass through the frozen target's matched-message and shuffled-message computations to update the bridge. Source generation, teacher predictions, and the zero-message and target-only conditions are computed without gradients. We use AdamW with learning rate 10^{-5} and weight decay 0.01. The physical batch size is two, with four gradient-accumulation steps, giving eight examples per optimizer update. Message shuffling takes place within each physical batch. Gradients are clipped to a maximum norm of 1.0.

End-to-end training is configured for ten epochs, with early stopping after four consecutive validation checks without improvement. We use a learning rate of 10^{-5}. Validation is performed every 250 optimizer updates and at the end of the epoch. Each validation pass uses the first 64 batches of two examples, corresponding to 128 examples from the validation partition. We retain the checkpoint with the lowest mean validation value of the complete training objective, rather than selecting checkpoints by benchmark accuracy.

D.1 Validation diagnostics of message specificity

We examine whether the receiver's predictions depend on receiving the message associated with its input. At a communication budget of $H = 80$, we compare the reference-response cross-entropy under correctly paired messages with three controls: messages swapped between examples, zero-valued messages, and the receiver without transferred message positions. The receiver's question and reference answer remain fixed across conditions. Figure 3 reports each control's loss minus the correct-message loss, measured in nats per token. Positive values therefore favor the correctly paired message. Rows identify source–target pairs, while columns identify their corresponding training and validation domains; these are within-domain comparisons. Together, these observations are consistent with useful, question-specific communication on the validation examples.

E Training-Free Vocabulary Alignment

E.1 Transfer Procedure

The training-free baseline transfers the source token weights and reconstructs each soft token using the target model's input embeddings. Correspondences are defined by exact matches between the token strings in the two tokenizer vocabularies.

Table 3: Shared vocabulary entries and the percentage of each model's vocabulary that they cover.

Source	Target	Shared entries	Source (%)	Target (%)
Qwen3-8B	Qwen3-8B	151,669	100.00	100.00
Qwen3-8B	Qwen3-4B	151,669	100.00	100.00
Qwen3-8B	Ministral	67,861	44.74	51.77
Gemma	Ministral	21,281	8.12	16.24
Qwen3-8B	Gemma	24,520	16.17	9.35

Let $\mathrm{tok}_A(i)$ and $\mathrm{tok}_B(j)$ denote the vocabulary strings associated with source token i and target token j. We define

$$\mathcal{M}_{A\to B} = \{(i,j) : \mathrm{tok}_A(i) = \mathrm{tok}_B(j)\}. \tag{22}$$

Let $m_{A\to B}(i)$ be the target token ID corresponding to source token i. At generation step t, the matched source tokens are

$$\mathcal{C}^t_{A\to B} = \left\{ i \in \mathcal{S}^t_A : \exists j \text{ such that } (i,j) \in \mathcal{M}_{A\to B} \right\}, \tag{23}$$

where $\mathcal{S}^t_A$ is the retained source token set. For CIPHER-Cross (Section M), this set is the full source vocabulary, and the weights are the CIPHER softmax probabilities. Their total weight is

$$Z^t_{A\to B} = \sum_{i\in\mathcal{C}^t_{A\to B}} w^t_{A,i}. \tag{24}$$

The transferred embedding is

$$\tilde{e}^t_B = \begin{cases} \dfrac{1}{Z^t_{A\to B}} \displaystyle\sum_{i\in\mathcal{C}^t_{A\to B}} w^t_{A,i} e^B_{m_{A\to B}(i)}, & Z^t_{A\to B} > 0, \\ \mathbf{0}_{d_B}, & Z^t_{A\to B} = 0. \end{cases} \tag{25}$$

We apply no additional norm rescaling.

The amount of information retained depends on which source tokens receive weight and whether they have target matches. If only one token with nonzero weight is matched, the transferred embedding reduces to that token's target embedding. If none are matched, the target receives a zero embedding. If every token with nonzero weight is matched, the procedure discards no weight.

For Gemma targets, this baseline supplies the reconstructed main embedding together with the padding token's native per-layer embeddings at each communication position.

E.2 Vocabulary Overlap

Table 3 reports the number of shared vocabulary entries and their coverage of each vocabulary. These percentages describe vocabulary overlap; the weight retained at a particular generation step is measured by $Z^t_{A\to B}$.

F Benchmark Details

F.1 Mathematical Reasoning

AIME 2024 and AIME 2025 AIME contains competition mathematics problems from the American Invitational Mathematics Examination. The problems require multiple reasoning steps and have integer-valued answers. The 2024 examination and 2025 examination each have 30 problems (Hugging Face H4, 2024; Zhang & Math-AI Team, 2025).

AMC 2023 AMC 2023 contains competition mathematics problems from the 2023 American Mathematics Competitions. It contains 40 problems, each paired with a reference answer (Math-AI, 2023).

MATH-500 MATH-500 is a subset of 500 problems from the MATH test set. It covers seven subjects including pre-algebra, algebra, intermediate algebra, number theory, geometry, precalculus, and counting and probability. The problems require deriving a mathematical answer from a written problem statement (Hendrycks et al., 2021; Lightman et al., 2024).

GSM8K GSM8K contains grade-school mathematics word problems that require a sequence of arithmetic operations. Each problem is accompanied by a worked solution and a final numerical answer. It contains 1319 problems (Cobbe et al., 2021).

F.2 Knowledge Tasks

OpenBookQA OpenBookQA contains four-option science questions that require combining elementary scientific facts with commonsense knowledge. Our evaluation set contains 500 questions (Mihaylov et al., 2018).

GPQA Diamond GPQA contains graduate-level multiple-choice questions written and validated by experts in biology, chemistry, and physics. We use the Diamond subset of 198 questions (Rein et al., 2023).

ARC-Challenge and ARC-Easy The AI2 Reasoning Challenge contains multiple-choice science questions from grade-school examinations. ARC-Challenge contains questions answered incorrectly by both a retrieval baseline and a word-co-occurrence baseline; ARC-Easy contains the remaining questions. ARC-Challenge has 1172 questions, and ARC-Easy has 2376 questions (Clark et al., 2018).

MedQA MedQA contains questions from medical licensing examinations that assess medical knowledge and clinical reasoning. We use the English USMLE version with four answer choices, which has 1273 questions (Jin et al., 2020).

F.3 Code Generation

HumanEval HumanEval contains 164 hand-written Python programming tasks. Each task provides a function signature and a docstring describing the required behavior. Generated solutions are evaluated for functional correctness using executable tests (Chen et al., 2021).

MBPP sanitized MBPP contains programming tasks covering basic algorithms and Python functionality. Each problem includes a natural-language task description, a reference solution, and automated tests. It contains 257 problems (Austin et al., 2021).

MBPP+ MBPP+ extends sanitized MBPP with additional test cases to check the functional correctness of generated programs. It contains 378 problems (Liu et al., 2023).

G LLM Judge Prompts

We use GPT-5.2 to check whether the model's answer agrees with the reference answer. The original question is not included. If the model response contains a closing `</think>` marker, we send the text after that marker to the judge. Otherwise, we send the full response. Text in square brackets is replaced with the response, reference answer, or answer extracted by the rule-based evaluator. The judge is asked to return `YES` or `NO`.

G.1 Mathematical Reasoning

AIME 2024, AIME 2025, AMC 2023, MATH-500, and GSM8K use the following prompt.

```
Please determine whether the final answer provided in the model-generated
response is equivalent to the reference answer from a math question. The
final answer may either be enclosed in \boxed{} or appear after
"Answer:". If they are equivalent, return "YES"; if they are not, return
"NO". Only return "YES" or "NO", and do not generate any other content.
Model-generated answer: [MODEL_RESPONSE]
Reference answer: [REFERENCE_ANSWER]
```

G.2 KNOWLEDGE TASKS

OpenBookQA, GPQA Diamond, ARC-Challenge, ARC-Easy, and MedQA use the following prompt. The judge also receives the answer extracted by the rule-based evaluator.

```
Please determine whether the final answer provided in the model-generated
response selects the same multiple-choice option as the reference answer.
The final answer may be written as a letter, number, option, or boxed
answer. If they are equivalent, return "YES"; if they are not, return
"NO". Only return "YES" or "NO", and do not generate any other content.
Model-generated answer: [MODEL_RESPONSE]
Rule-extracted answer: [RULE_EXTRACTED_ANSWER]
Reference answer: [REFERENCE_ANSWER]
```

H AGENT PROMPT TEMPLATES

H.1 MATHEMATICAL REASONING

Source instruction

```
Let's first understand the problem and make a complete plan.
```

Target instruction

```
Let's carry out the plan. Please reason step by step, and put your final
answer within \boxed{{}}.
```

H.2 KNOWLEDGE TASKS

Source instruction

```
Let's first understand the multiple-choice question. Identify the
relevant facts, principles, and answer choices. Make a complete plan.
```

Target instruction

```
Let's carry out the plan. Reason carefully and verify that the answer
matches the relevant facts. Put the selected option label within
\boxed{}, with nothing else inside the box
```

H.3 CODE GENERATION

Source instruction.

```
Let's first understand the programming task. Identify the relevant
algorithm, data structures, and interface requirements. Make a complete
plan.
```

Target instruction.

````
Let's carry out the plan. Reason carefully and verify that the solution
satisfies the interface requirements. Return the complete solution,
including necessary imports and helpers, in exactly one Python code
block.

``python
# Complete Python solution, including required imports, functions, and
helpers.
```
````

H.4 Task Input Formatting

The templates below show how task inputs are formatted as user messages. Text in square brackets is replaced with the question, answer choices, or programming task for each example.

Mathematical reasoning The user message contains the original problem statement, without an additional fixed instruction.

```
[PROBLEM_STATEMENT]
```

Knowledge tasks The user message contains the question and its labeled answer choices.

```
[QUESTION]

[LABEL_1]. [OPTION_1]
...
[LABEL_K]. [OPTION_K]
```

Code generation For HumanEval and MBPP sanitized, the user message contains the original dataset prompt. HumanEval prompts include a function signature and a specification of the required behavior. MBPP sanitized contains a description of a programming task.

HumanEval

```
[ORIGINAL_HUMANEVAL_PROMPT]
```

MBPP sanitized

```
[ORIGINAL_MBPP_SANITIZED_PROMPT]
```

MBPP+ The user message adds a code-format instruction and up to three visible test assertions to the problem statement.

````
Please provide a self-contained Python script that solves the following
problem in a markdown code block:
```python
YOUR_PYTHON_CODE
```

[PROBLEM_STATEMENT]

Your solution should satisfy assertions like:
[VISIBLE_ASSERTION_1]
[VISIBLE_ASSERTION_2]
[VISIBLE_ASSERTION_3]
````

I Reasoning Configuration

We set `enable_thinking=True` when applying the chat templates for Qwen3-8B, Qwen3-4B, and gemma-4-E4B-it wherever these models serve as source or target agents, including single-agent evaluations. For Ministral-3-3B-Reasoning-2512, we prepend a model-specific system instruction
````

requesting a thinking phase delimited by `[THINK]` and `[/THINK]`, followed by the response; no separate `enable_thinking` argument is supplied. These settings are retained across the compared communication methods. We evaluate all models and baselines using three random seeds (42, 123, and 21) and report mean accuracy and sample standard deviation across seeds.

J Computation Details

Bridge training was performed on NVIDIA RTX A6000 GPUs, and evaluation was performed on NVIDIA L40S GPUs. During task-level bridge training, the frozen language models use BF16, while the bridge parameters are maintained in FP32. Gradient checkpointing is enabled in the target model. During evaluation, the language models are loaded in BF16. We use the FlashAttention 2 backend for Qwen and the PyTorch scaled dot-product attention (SDPA) backend for Ministral and Gemma.

K Inference timing

We measure average inference time per sample as total timed batch duration divided by the number of evaluated examples, using batches of eight with CUDA synchronization before and after each batch. Timing includes prompt processing, source generation, message transfer, target generation, and CPU–GPU model transfers, including offloading. It excludes initial checkpoint loading, dataset loading, output saving, and answer scoring, including LLM judging.

L Same-Model Baseline Details

CIPHER. CIPHER was introduced for multiagent debate. We use its soft-token generation mechanism within our source–target setting: the source generates a continuous message, and the target uses this message to produce its response, without further debate rounds. At each source step, we compute a probability distribution over the full vocabulary using a softmax temperature of 0.6 and form the next input as the probability-weighted sum of input embeddings. The source generates H continuous positions, which are passed directly to the target.

LatentMAS. Our adaptation generates H latent thoughts by mapping the source's final-layer hidden state into its input-embedding space and feeding the mapped vector back into the model at each step. For transfer, we map the collected hidden states into the target's input-embedding space and rescale each vector to the mean norm of the target's token embeddings. The target receives these vectors as additional input positions.

M Cross-Model CIPHER Baseline

Source generation. CIPHER-Cross combines the CIPHER source-generation procedure in Section L with the training-free vocabulary alignment in Section E. At each source step, we compute

$$p^t_{A,i} = \frac{\exp(z^t_{A,i}/0.6)}{\sum_{j\in\mathcal{V}_A}\exp(z^t_{A,j}/0.6)}, \qquad e^t_A = \sum_{i\in\mathcal{V}_A} p^t_{A,i} e^A_i, \tag{26}$$

where z^t_A denotes the source logits and $\mathcal{V}_A$ is the full source vocabulary. We feed e^t_A back to the source and retain the probability vector for transfer. Unlike the stochastic training-free baseline, CIPHER-Cross applies no top-k or top-p truncation, Dirichlet sampling, or Gumbel–Softmax sampling.

Cross-model transfer. We apply eq. (25) with $w^t_{A,i} = p^t_{A,i}$ and $\mathcal{S}^t_A = \mathcal{V}_A$. We retain exact token-string matches, renormalize their probability mass, and reconstruct the message using the corresponding target embeddings. If the matched mass is zero, we use a zero embedding. We apply no additional norm rescaling. Both models remain frozen, and the transfer uses neither paired training data nor learned parameters. This is our cross-model adaptation of CIPHER, rather than a learned CIPHER bridge.

Table 4: Additional cross-model communication across four source–target pairs, accuracy ($M \pm SD$, 3 seeds) and latency (s/sample). Bold marks the best displayed value across this table and Table 2, excluding full text and including ties. DIALECT uses $H = 80$; C2C uses prompt-prefill cache transfer without a matched thought budget.

Method	Metric	AIME 2024	AIME 2025	AMC 2023	MATH 500	GSM 8K	Open Book	GPQA Diamond	ARC C	ARC E	Med QA	Human Eval	MBPP	MBPP +
Qwen3-8B → Qwen3-4B														
CIPHER-Cross	Acc.	$72.2_{\pm 5.1}$	$50.0_{\pm 3.3}$	$93.3_{\pm 3.8}$	$92.9_{\pm 1.1}$	$92.5_{\pm 0.7}$	$92.3_{\pm 1.2}$	$51.5_{\pm 3.1}$	$93.8_{\pm 0.7}$	$95.6_{\pm 0.4}$	$72.1_{\pm 1.3}$	$88.2_{\pm 2.5}$	$77.8_{\pm 2.6}$	$76.7_{\pm 2.2}$
	Latency	321.8	220.9	107.1	82.2	27.1	6.3	151.1	26.0	17.1	59.5	53.5	39.9	80.2
C2C	Acc.	$53.3_{\pm 3.3}$	$50.0_{\pm 3.3}$	$87.5_{\pm 2.5}$	$94.0_{\pm 0.8}$	$93.2_{\pm 0.8}$	$95.0_{\pm 0.2}$	$35.2_{\pm 1.8}$	$92.9_{\pm 0.6}$	$96.5_{\pm 0.5}$	$60.9_{\pm 0.6}$	$87.2_{\pm 1.2}$	$79.6_{\pm 1.4}$	$69.8_{\pm 1.1}$
	Latency	306.3	**209.3**	101.0	77.3	26.4	6.1	**139.3**	26.3	16.6	57.5	50.5	40.6	76.3
DIALECT	Acc.	**73.3**$_{\pm 3.3}$	**60.0**$_{\pm 3.3}$	**97.5**$_{\pm 2.5}$	**96.0**$_{\pm 0.8}$	**95.5**$_{\pm 0.5}$	**95.4**$_{\pm 0.8}$	$56.7_{\pm 2.8}$	**95.3**$_{\pm 0.7}$	**97.7**$_{\pm 0.4}$	**73.9**$_{\pm 0.4}$	**91.3**$_{\pm 1.4}$	**86.4**$_{\pm 1.8}$	**79.9**$_{\pm 1.2}$
	Latency	**286.1**	216.0	**98.1**	77.2	**25.3**	**5.7**	140.4	**24.3**	**16.3**	**57.2**	**50.0**	**38.4**	**75.6**
Qwen3-8B → Gemma-4-E4B-it														
CIPHER-Cross	Acc.	$44.4_{\pm 9.6}$	$41.1_{\pm 8.4}$	$77.5_{\pm 6.6}$	$84.1_{\pm 1.6}$	$81.2_{\pm 1.1}$	$87.1_{\pm 1.5}$	$55.2_{\pm 3.5}$	$88.4_{\pm 0.9}$	$96.1_{\pm 0.4}$	$70.4_{\pm 1.3}$	$72.2_{\pm 3.5}$	$65.9_{\pm 3.0}$	$38.4_{\pm 2.5}$
	Latency	244.5	216.8	204.8	69.5	21.2	22.6	53.2	21.0	17.4	75.9	116.5	40.3	88.5
C2C	Acc.	$56.7_{\pm 3.3}$	$40.0_{\pm 3.3}$	$85.0_{\pm 2.5}$	$86.5_{\pm 0.9}$	$91.5_{\pm 0.8}$	$88.0_{\pm 1.0}$	$35.5_{\pm 1.8}$	$93.7_{\pm 0.6}$	$98.2_{\pm 0.5}$	$61.9_{\pm 0.6}$	$78.9_{\pm 1.5}$	$66.7_{\pm 1.4}$	$51.9_{\pm 1.1}$
	Latency	229.5	202.6	197.9	66.9	20.3	**20.9**	**50.8**	20.0	16.3	72.3	112.7	37.6	84.6
DIALECT	Acc.	**63.3**$_{\pm 3.3}$	**60.0**$_{\pm 3.3}$	**95.0**$_{\pm 4.3}$	**91.0**$_{\pm 1.0}$	**94.3**$_{\pm 0.5}$	**92.0**$_{\pm 0.8}$	$56.9_{\pm 1.5}$	**95.8**$_{\pm 0.6}$	**98.8**$_{\pm 0.4}$	**74.6**$_{\pm 0.5}$	**85.8**$_{\pm 1.8}$	**77.8**$_{\pm 1.0}$	$67.9_{\pm 0.9}$
	Latency	**209.4**	**198.3**	197.2	**64.3**	**18.6**	22.3	51.0	**15.4**	**16.2**	**63.7**	**89.2**	**36.3**	**81.5**
Qwen3-8B → Ministral-3B														
CIPHER-Cross	Acc.	$52.2_{\pm 9.6}$	$38.9_{\pm 8.4}$	$88.3_{\pm 5.2}$	$90.1_{\pm 1.3}$	$91.0_{\pm 0.8}$	$87.3_{\pm 1.5}$	$46.1_{\pm 3.5}$	$89.5_{\pm 0.9}$	$95.0_{\pm 0.5}$	$57.8_{\pm 1.4}$	$85.6_{\pm 2.7}$	$89.0_{\pm 2.0}$	$66.9_{\pm 2.4}$
	Latency	250.9	246.8	137.5	100.2	23.1	19.3	139.0	25.6	17.6	65.7	81.5	42.4	80.1
C2C	Acc.	$36.7_{\pm 3.3}$	$28.9_{\pm 5.1}$	$86.7_{\pm 1.4}$	$94.3_{\pm 0.9}$	$91.8_{\pm 0.8}$	$88.0_{\pm 1.0}$	$38.9_{\pm 1.5}$	$89.1_{\pm 0.6}$	$96.4_{\pm 0.5}$	$49.7_{\pm 0.7}$	$86.4_{\pm 1.5}$	$88.5_{\pm 1.4}$	$62.8_{\pm 1.2}$
	Latency	236.7	231.8	133.5	96.8	21.6	17.9	133.7	30.0	18.9	65.5	76.9	39.8	77.0
DIALECT	Acc.	**56.7**$_{\pm 3.3}$	**43.3**$_{\pm 3.3}$	$89.2_{\pm 3.8}$	**95.4**$_{\pm 0.8}$	**93.4**$_{\pm 0.5}$	**90.6**$_{\pm 0.8}$	**48.0**$_{\pm 1.3}$	**91.0**$_{\pm 0.6}$	**97.1**$_{\pm 0.4}$	$59.6_{\pm 0.5}$	**89.0**$_{\pm 2.1}$	**90.7**$_{\pm 1.4}$	**70.1**$_{\pm 0.9}$
	Latency	**201.0**	**220.8**	**132.2**	96.1	**20.3**	**15.7**	**119.1**	**24.4**	**16.4**	**62.0**	**71.9**	**39.0**	**75.8**
Gemma-4-E4B-it → Ministral-3B														
CIPHER-Cross	Acc.	$31.1_{\pm 8.4}$	$34.4_{\pm 8.4}$	$64.2_{\pm 7.6}$	$70.1_{\pm 2.1}$	$68.1_{\pm 1.3}$	$30.3_{\pm 2.1}$	$25.8_{\pm 3.1}$	$88.4_{\pm 0.9}$	$95.7_{\pm 0.4}$	$58.2_{\pm 1.4}$	$41.7_{\pm 3.9}$	$50.8_{\pm 3.1}$	$37.8_{\pm 2.5}$
	Latency	231.3	266.8	215.2	104.9	20.0	23.7	112.5	26.5	17.5	67.7	103.7	52.2	68.5
C2C	Acc.	$36.7_{\pm 3.3}$	$12.2_{\pm 5.1}$	$85.0_{\pm 2.5}$	$93.3_{\pm 0.9}$	$89.8_{\pm 0.8}$	$85.4_{\pm 1.0}$	$13.8_{\pm 1.8}$	$87.0_{\pm 0.6}$	$95.9_{\pm 0.5}$	$40.7_{\pm 0.7}$	$84.8_{\pm 1.2}$	$85.3_{\pm 1.4}$	$54.3_{\pm 1.2}$
	Latency	223.5	256.6	206.0	97.1	19.1	22.6	106.6	27.3	17.4	74.6	96.5	50.2	65.0
DIALECT	Acc.	**56.7**$_{\pm 3.3}$	**40.0**$_{\pm 3.3}$	**90.0**$_{\pm 4.3}$	**95.4**$_{\pm 0.9}$	**93.0**$_{\pm 0.5}$	**90.0**$_{\pm 0.8}$	**40.9**$_{\pm 1.5}$	**91.1**$_{\pm 0.6}$	**97.2**$_{\pm 0.4}$	**59.4**$_{\pm 0.5}$	**89.6**$_{\pm 1.6}$	**90.0**$_{\pm 1.1}$	**68.7**$_{\pm 1.1}$
	Latency	**200.3**	**203.5**	**177.2**	**70.4**	**14.4**	**10.9**	**97.5**	**24.8**	**16.7**	**65.0**	**91.3**	**28.7**	**49.0**

Evaluation and interpretation. We use $H = 80$ continuous positions, the same planning-and-solving prompts, and the target decoding settings in Section 4.5. Table 4 compares CIPHER-Cross and DIALECT on the four cross-model pairs. The stochastic training-free baseline shares DIALECT's source generation and changes the transfer procedure; CIPHER-Cross changes both source generation and transfer relative to DIALECT. Comparing the two training-free baselines keeps vocabulary transfer fixed while changing source generation. Statistical comparisons follow Section P.

N CACHE-TO-CACHE BASELINE

We adapt C2C (Fu et al., 2026) to our model pairs and evaluation pipeline, training a separate cache projector for each pair and domain while freezing both language models. Our implementation transfers source prompt-prefill caches without autoregressive source generation. Its cache size depends on the input and model architecture, rather than an H-position thought budget.

Prompts. For C2C, both models receive the target's system instruction and the same task input, formatted using their respective chat templates. The source supplies prompt-prefill caches without generating a separate plan.

Training. We train each cache projector on the corresponding domain of the Llama-Nemotron-Post-Training-Dataset (Bercovich et al., 2025), keeping both language models frozen. We use

AdamW with a peak learning rate of 10^{-4}, weight decay 0.01, and gradient-norm clipping at 1.0. Training makes one pass through the retained training examples, using an effective batch size of 128 and a smaller final batch, for 113 optimizer updates. The learning rate increases linearly over the first 11 updates and then follows a linear decay schedule. Training uses seed 42.

Checkpoint selection. We evaluate generated-answer accuracy at updates 1, 25, 50, 75, 100, and 113, retaining the earliest checkpoint attaining the highest validation accuracy. Selection uses fixed subsets of the original validation split: 16 reviewed math examples, 64 knowledge examples, and 64 code examples. These subsets exclude training examples and benchmark questions. The math selection subset is a diagnostic cohort chosen for independently checkable answers, rather than a representative estimate of math accuracy. The released manifests specify the math and code selection examples and grading information; the knowledge selection subset is specified by its offset and size within the frozen validation split.

Math cohort exclusions. For C2C math training, the frozen exclusion manifest removes four training examples and ten validation examples with identified reference defects, leaving 14,396 training examples. The 16-example checkpoint-selection cohort is separate from the eight reviewed validation examples reserved for the final quality check.

O Qualitative Analysis of LatentMAS Thoughts

To investigate the weaker LatentMAS performance observed in our evaluation, we inspect its generated thoughts and the responses produced from them. We use three settings: (i) greedily decoding the thought representations; (ii) asking the model to reconstruct the plan from the thoughts without the original question; and (iii) asking it to reconstruct the plan with both the thoughts and the original question. Table 5 shows three illustrative examples from GSM8K. Long repetitive outputs are shortened for readability.

Greedy decoding produces repeated tokens or fragments across consecutive thought steps. In the first two examples, the decoded sequences contain repeated punctuation and programming-related terms rather than a readable mathematical plan. In the third example, the sequence includes a task-related word but does not reveal a complete solution.

Without the original question, reconstruction does not produce a useful plan in these examples. The model instead repeats the reconstruction instruction, requests clarification, or returns only punctuation. With the question included, it produces correct solutions for the first two examples but still returns an uninformative response for the third. Because the question itself contains the information needed to solve the problem, a correct response in this setting does not establish that the model used the communicated thought.

Together, these observations suggest that LatentMAS thoughts may become highly similar across steps and may not reliably convey independently decodable information to subsequent agents.

P Statistical Comparisons with Baselines

Observations. We compare DIALECT with each baseline across 13 benchmarks, separately for each source–target pair with DIALECT at $H = 80$. We use all three seed-level results for each benchmark, giving 39 observations paired by benchmark and seed. Seeds are not averaged before testing. Mean accuracy differences and speedups summarize the results; the signed-rank tests use the individual paired differences.

Accuracy. For each benchmark and seed, we calculate DIALECT accuracy minus baseline accuracy in percentage points. Positive differences favor DIALECT. We use one-sided Wilcoxon signed-rank tests to assess accuracy improvements over the single-agent target, budget-matched text transfer, CIPHER (Pham et al., 2024), LatentMAS (Zou et al., 2025), training-free alignment, CIPHER-Cross, and C2C (Fu et al., 2026). These tests assess the location of paired differences under the signed-rank assumptions, rather than the mean difference.

Table 5: Qualitative comparison of thought decoding methods on three samples. Long repetitive outputs are truncated for readability.

ID	Method	Question / Generated Output
0	Question	Janet's ducks lay 16 eggs per day. She eats three for breakfast every morning and bakes muffins for her friends every day with four. She sells the remainder at the farmers' market daily for $2 per fresh duck egg. How much in dollars does she make every day at the farmers' market?
	Greedy thought decoding	`<think>. (((matplotlib. ..(. sys.....(( React. ... if if if if if ...`
	Thought repetition	Janet's ducks lay 16 eggs per day. She eats 3 eggs for breakfast and uses 4 eggs to bake muffins. Therefore, $16 - 3 - 4 = 9$ eggs remain. At $2 per egg, she makes $9 \times 2 = \$18$ per day.
	Thought repetition without question	The supplied internal representations contain a solution plan. Repeat all information contained in them faithfully and clearly in text.
1	Question	A robe takes 2 bolts of blue fiber and half that much white fiber. How many bolts in total does it take?
	Greedy thought decoding	`<think>. (((matplotlib. .(((matplotlib. ... if if if if if ...`
	Thought repetition	A robe takes 2 bolts of blue fiber and half that much white fiber. Half of 2 bolts is 1 bolt. Thus, the robe takes 2 bolts of blue fiber and 1 bolt of white fiber, for a total of $2 + 1 = 3$ bolts.
	Thought repetition without question	Okay, let me try to understand what you're asking. It seems like you're providing a series of repeated phrases, possibly related to a technical or programming context, but the meaning isn't clear. Could you please clarify what you need help with?
2	Question	Josh decides to try flipping a house. He buys a house for $80,000 and then puts in $50,000 in repairs. This increased the value of the house by 150%. How much profit did he make?
	Greedy thought decoding	`<think>. profit</think> ...`
	Thought repetition	.
	Thought repetition without question	.

Latency. For each benchmark and seed, we calculate the log ratio of baseline latency to DIALECT latency. Positive values favor DIALECT. We use a one-sided Wilcoxon signed-rank test to assess latency reductions and report the geometric mean of the seed-level speedups:

$$S_{\text{geo}} = \exp\left(\frac{1}{39}\sum_{b=1}^{13}\sum_{s=1}^{3}\log\frac{T_{\text{baseline},bs}}{T_{\text{DIALECT},bs}}\right).$$

This summary gives each benchmark equal weight. Values above one favor DIALECT; a speedup of $2\times$ corresponds to a geometric-mean DIALECT-to-baseline latency ratio of 0.5. Single-agent latency comparisons are omitted because these measurements are not available.

Signed-rank calculations. We exclude zero differences and assign average ranks to tied absolute differences. We report W^+, the sum of ranks for positive differences. Differences are rounded to six decimal places before ranking to avoid numerical artifacts. We calculate exact conditional p-values by enumerating the signed-rank distribution using dynamic programming. Although each

benchmark contributes three observations, its contribution to the test depends on the ranks of those differences. Tables 6 and 7 report the full results.

Accuracy retention. At $H = 80$, let $\bar{A}_{m,b}$ denote the accuracy of method m on benchmark b, averaged over three seeds. We report

$$R = 100 \times \frac{\sum_{b=1}^{13} \bar{A}_{\mathrm{DIALECT},b}}{\sum_{b=1}^{13} \bar{A}_{\mathrm{Full},b}}.$$

This ratio weights all benchmarks equally and differs from averaging benchmark-specific accuracy ratios. It summarizes observed accuracy retention; statistical equivalence is assessed separately below.

Full-text accuracy equivalence. Against full-text communication, we use paired Wilcoxon-based TOST with bounds of ± 2 percentage points. Let d_{bs} denote DIALECT accuracy minus full-text accuracy, and let θ denote the center (location) of the paired-difference distribution under symmetry. The two tests assess

$$H_{0,L} : \theta \leq -2 \quad \text{against} \quad H_{1,L} : \theta > -2,$$
$$H_{0,U} : \theta \geq 2 \quad \text{against} \quad H_{1,U} : \theta < 2.$$

We apply a one-sided signed-rank test to $d_{bs} + 2$ in the positive direction and to $d_{bs} - 2$ in the negative direction. Both tests use all 39 seed-level pairs. Ranks are recalculated separately at each bound using the procedure above. The reported TOST p-value is the larger of the two component p-values:

$$p_{\mathrm{TOST}} = \max(p_L, p_U).$$

Equivalence is supported only when both component p-values are below .05. No adjustment is needed between these two component tests because both must reject for equivalence to be established.

We report the Hodges–Lehmann estimate of the paired differences, calculated as the median of the 780 values $(d_i + d_j)/2$ for $1 \leq i \leq j \leq 39$. This estimates the pseudomedian, which equals the location under symmetry. The equivalence criterion concerns this location, not equality of means or equivalence on every benchmark. Failure to establish equivalence does not by itself establish a practically meaningful difference. Table 8 reports both component tests and the TOST results.

Multiple comparisons. We apply Holm correction separately within each source–target pair for accuracy-superiority and latency-reduction comparisons, using a significance level of .05. Each family contains four comparisons in the same-model setting and five in each cross-model setting after adding C2C. The cross-model accuracy family includes Single, Text (80), Training-free, CIPHER-Cross, and C2C; the latency family includes Text (80), Training-free, CIPHER-Cross, C2C, and Full text. We apply the same Holm correction families separately to the primary signed-rank tests and the benchmark-clustered sensitivity analysis. Each pair's single full-text equivalence comparison is assessed separately without additional correction. This correction applies within the defined families, not across all claims in the paper, and does not account for dependence within benchmarks.

Assumptions and sensitivity. Our primary analysis uses one-sided Wilcoxon signed-rank tests on the 39 benchmark–seed paired differences, retaining individual seed-level observations. The tests assume independent paired differences and symmetry for their location-based interpretation. As a supplementary sensitivity analysis, we fit a regression to the same 39 differences with seed fixed effects and covariance estimates clustered by benchmark. We use a finite-sample correction and a t reference distribution with 12 degrees of freedom, applying Holm correction within the same comparison families. This analysis allows dependence within benchmarks without averaging across seeds. Table 9 compares its results with those of the primary signed-rank analysis. The regression assesses mean contrasts rather than the signed-rank location and therefore does not provide a sensitivity test of the equivalence claim in Table 8.

Results. Under the signed-rank assumptions, accuracy improvements and latency reductions are statistically significant across all 24 comparisons for each outcome after within-pair correction. Against full text, Wilcoxon-based TOST supports equivalence within ± 2 percentage points for the center of the paired accuracy differences in all five model pairs.

Table 6: Accuracy comparisons with DIALECT using one-sided Wilcoxon signed-rank tests. Holm correction is applied within model pair and outcome.

Baseline	Mean gain (pp)	$n_{\neq 0}$	W^+	Raw p	Holm p
Qwen3-8B → Qwen3-8B					
Single	+3.54	38	741	$< .001$	$< .001$
Text (80)	+1.40	34	537	$< .001$	$< .001$
CIPHER	+2.89	37	656.5	$< .001$	$< .001$
LatentMAS	+20.24	39	780	$< .001$	$< .001$
Qwen3-8B → Qwen3-4B					
Single	+2.69	38	741	$< .001$	$< .001$
Text (80)	+4.42	39	780	$< .001$	$< .001$
Training-free	+1.24	36	549.5	$< .001$	$< .001$
CIPHER-Cross	+3.84	37	703	$< .001$	$< .001$
C2C	+7.99	39	779	$< .001$	$< .001$
Qwen3-8B → Gemma-4-E4B-it					
Single	+5.49	36	661	$< .001$	$< .001$
Text (80)	+7.00	39	754	$< .001$	$< .001$
Training-free	+9.81	39	776	$< .001$	$< .001$
CIPHER-Cross	+11.64	38	741	$< .001$	$< .001$
C2C	+9.15	39	780	$< .001$	$< .001$
Qwen3-8B → Ministral-3B					
Single	+2.66	37	703	$< .001$	$< .001$
Text (80)	+3.46	38	741	$< .001$	$< .001$
Training-free	+0.91	32	429.5	$< .001$	$< .001$
CIPHER-Cross	+2.79	36	640.5	$< .001$	$< .001$
C2C	+5.85	38	741	$< .001$	$< .001$
Gemma-4-E4B-it → Ministral-3B					
Single	+1.73	35	625	$< .001$	$< .001$
Text (80)	+3.31	39	780	$< .001$	$< .001$
Training-free	+21.77	39	780	$< .001$	$< .001$
CIPHER-Cross	+23.50	38	741	$< .001$	$< .001$
C2C	+10.60	39	780	$< .001$	$< .001$

Q Additional Results Across Communication Budgets

Figure 4 presents the accuracy–latency trade-offs on the remaining ten benchmarks across all five source–target pairs. These results complement those in the main text.

R Accuracy–Latency Regression

We analyze all individual seed-level observations across five source–target pairs, 13 benchmarks, three budgets, and three seeds. Seeds are not averaged before fitting. For observation i in pair–benchmark group g, we fit

$$\operatorname{logit}(\widetilde{p}_{ig}) = \alpha + \beta \left(\log T_{ig} - \overline{\log T}_g\right) + u_g + \epsilon_{ig},$$

where p_{ig} denotes accuracy, T_{ig} denotes latency, and u_g is a group-specific random intercept. Centering log latency isolates the within-setting association. To obtain finite logits, we clip accuracy to $[10^{-4}, 1 - 10^{-4}]$, affecting only three observations with accuracy equal to one.

We estimate the model using restricted maximum likelihood and report a two-sided Wald test and a 95% Wald confidence interval. The estimated slope is positive ($\beta = 1.40$, 95% CI $[0.76, 2.04]$, $p < .001$). Clipping thresholds of 10^{-3} and 10^{-5} give slopes of 1.21 and 1.59, respectively, with $p < .001$ in both cases.

Table 7: Latency comparisons with DIALECT using one-sided Wilcoxon signed-rank tests. Speedups are geometric means; Holm correction is applied within model pair and outcome.

Baseline	Speedup	$n_{\neq 0}$	W^+	Raw p	Holm p
Qwen3-8B → Qwen3-8B					
Text (80)	1.34×	39	780	$< .001$	$< .001$
CIPHER	1.10×	39	705	$< .001$	$< .001$
LatentMAS	1.40×	39	780	$< .001$	$< .001$
Full text	2.24×	39	780	$< .001$	$< .001$
Qwen3-8B → Qwen3-4B					
Text (80)	1.33×	39	780	$< .001$	$< .001$
Training-free	1.03×	33	504	$< .001$	$< .001$
CIPHER-Cross	1.07×	39	780	$< .001$	$< .001$
C2C	1.03×	39	687	$< .001$	$< .001$
Full text	2.13×	39	780	$< .001$	$< .001$
Qwen3-8B → Gemma-4-E4B-it					
Text (80)	1.28×	39	780	$< .001$	$< .001$
Training-free	1.07×	36	600	$< .001$	$< .001$
CIPHER-Cross	1.13×	39	780	$< .001$	$< .001$
C2C	1.07×	39	696	$< .001$	$< .001$
Full text	1.90×	39	780	$< .001$	$< .001$
Qwen3-8B → Ministral-3B					
Text (80)	1.29×	39	780	$< .001$	$< .001$
Training-free	1.08×	36	660	$< .001$	$< .001$
CIPHER-Cross	1.11×	39	780	$< .001$	$< .001$
C2C	1.08×	39	780	$< .001$	$< .001$
Full text	1.99×	39	780	$< .001$	$< .001$
Gemma-4-E4B-it → Ministral-3B					
Text (80)	1.41×	39	780	$< .001$	$< .001$
Training-free	1.26×	39	780	$< .001$	$< .001$
CIPHER-Cross	1.31×	39	780	$< .001$	$< .001$
C2C	1.27×	39	780	$< .001$	$< .001$
Full text	2.02×	39	780	$< .001$	$< .001$

Table 8: Accuracy equivalence to full text using paired Wilcoxon-based TOST with ± 2-percentage-point bounds. HL denotes the Hodges–Lehmann estimate.

Source–target pair	HL (pp)	W_L^+	W_U^+	p_L	p_U	TOST p
Qwen3-8B → Qwen3-8B	-0.52	582	1	.0032	$< .001$	.0032
Qwen3-8B → Qwen3-4B	-0.60	661	100	$< .001$	$< .001$	$< .001$
Qwen3-8B → Gemma	-0.30	716.5	72	$< .001$	$< .001$	$< .001$
Qwen3-8B → Ministral	-0.74	609	1	$< .001$	$< .001$	$< .001$
Gemma → Ministral	-1.27	567.5	0	.0018	$< .001$	.0018

Using uncentered log latency instead gives a negative slope ($\beta = -0.49$, 95% CI $[-0.77, -0.21]$, $p < .001$). That specification combines within-group and between-group information under its random-effects assumptions. We use the centered specification to address changes within the same setting.

The model assumes Gaussian random intercepts and independent, constant-variance Gaussian residuals. Additional dependence across related benchmarks or matched runs is not modeled. The result describes an association rather than a causal effect of latency.

Table 9: Seed-level sensitivity analysis using signed-rank (SR) and benchmark-clustered (BC) comparisons. All p-values are Holm-adjusted within model pair and outcome.

Baseline	Accuracy SR p	Accuracy BC p	Latency SR p	Latency BC p
Qwen3-8B → Qwen3-8B				
Single	$< .001$	.0292	—	—
Text (80)	$< .001$	.0377	$< .001$	$< .001$
CIPHER	$< .001$	.0329	$< .001$	.0041
LatentMAS	$< .001$	$< .001$	$< .001$	$< .001$
Full text	—	—	$< .001$	$< .001$
Qwen3-8B → Qwen3-4B				
Single	$< .001$	.0269	—	—
Text (80)	$< .001$	.0269	$< .001$	$< .001$
Training-free	$< .001$	.0359	$< .001$	.0166
CIPHER-Cross	$< .001$	$< .001$	$< .001$	$< .001$
C2C	$< .001$	.0035	$< .001$	.0166
Full text	—	—	$< .001$	$< .001$
Qwen3-8B → Gemma-4-E4B-it				
Single	$< .001$	.0071	—	—
Text (80)	$< .001$	.0071	$< .001$	.0019
Training-free	$< .001$	.0017	$< .001$	.0255
CIPHER-Cross	$< .001$	$< .001$	$< .001$	$< .001$
C2C	$< .001$	$< .001$	$< .001$	.0255
Full text	—	—	$< .001$	$< .001$
Qwen3-8B → Ministral-3B				
Single	$< .001$	.0145	—	—
Text (80)	$< .001$	.0145	$< .001$	$< .001$
Training-free	$< .001$	.0225	$< .001$	$< .001$
CIPHER-Cross	$< .001$	$< .001$	$< .001$	$< .001$
C2C	$< .001$	.0101	$< .001$	$< .001$
Full text	—	—	$< .001$	$< .001$
Gemma-4-E4B-it → Ministral-3B				
Single	$< .001$	.0149	—	—
Text (80)	$< .001$	.0149	$< .001$	$< .001$
Training-free	$< .001$	.0023	$< .001$	.0018
CIPHER-Cross	$< .001$	.0018	$< .001$	.0018
C2C	$< .001$	.0035	$< .001$	.0018
Full text	—	—	$< .001$	$< .001$

S Statistical Procedure for Budget Comparisons

We compare adjacent budgets separately for each source–target pair using all three seed-level results across 13 benchmarks. Each comparison contains 39 observations paired by benchmark and seed. Seeds are not averaged before testing.

For accuracy, we calculate the higher-budget result minus the lower-budget result in percentage points. For latency, we calculate the log ratio of higher-budget to lower-budget latency. One-sided Wilcoxon signed-rank tests assess whether these differences are positive. These tests assess the location of paired differences under the signed-rank assumptions, rather than their mean.

We round differences to six decimal places, exclude zero differences, and assign average ranks to tied absolute differences. Exact conditional p-values are calculated from the signed-rank distribution using dynamic programming. Holm correction is applied jointly across the four tests within each model pair: accuracy and latency for $H = 40$ versus $H = 80$ and $H = 80$ versus $H = 160$.

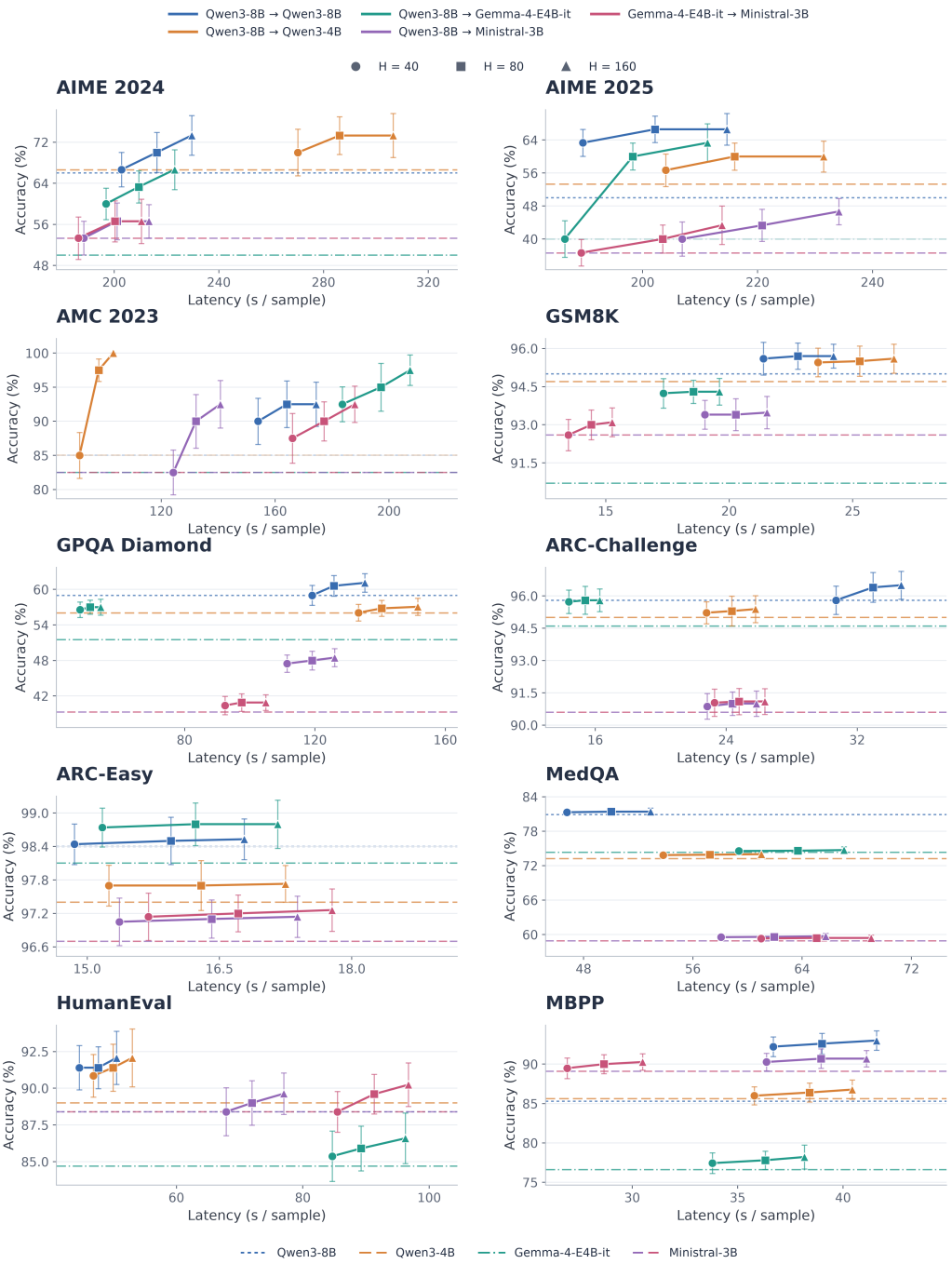


Figure 4: Accuracy–latency trade-offs for DIALECT across five pairs with communication budgets $H \in \{40, 80, 160\}$ (M $\pm$ SD, 3 seeds). Horizontal lines indicate single-agent target accuracy.

We report descriptive mean accuracy differences and geometric-mean latency increases calculated from the individual paired observations. Accuracy is used on its original scale, without the boundary adjustment required by the regression.

Table 10: Adjacent-budget comparisons using one-sided Wilcoxon signed-rank tests. Accuracy gains are in percentage points; latency increases are geometric-mean percentages. Holm correction is applied within each model pair.

Budget	Mean accuracy gain (pp)	Holm-adjusted p	Geometric-mean latency increase	Holm-adjusted p
Qwen3-8B → Qwen3-8B				
$40 \to 80$	$+0.97$	.0111	6.65%	$< .001$
$80 \to 160$	$+0.44$	.0478	6.14%	$< .001$
Qwen3-8B → Qwen3-4B				
$40 \to 80$	$+1.71$	.0052	6.49%	$< .001$
$80 \to 160$	$+0.33$	.0771	6.36%	$< .001$
Qwen3-8B → Gemma-4-E4B-it				
$40 \to 80$	$+2.19$	.0041	6.43%	$< .001$
$80 \to 160$	$+0.88$	.0155	6.72%	$< .001$
Qwen3-8B → Ministral-3B				
$40 \to 80$	$+1.27$	.1423	6.58%	$< .001$
$80 \to 160$	$+0.58$	.1861	6.23%	$< .001$
Gemma-4-E4B-it → Ministral-3B				
$40 \to 80$	$+0.95$	.0372	6.52%	$< .001$
$80 \to 160$	$+0.59$	.0437	6.33%	$< .001$

As in the baseline signed-rank analysis, these tests assume independent, symmetric paired differences and do not account for benchmark clustering. Their p-values should be interpreted under those assumptions.

T Results Across Adjacent Communication Budgets

Table 10 reports the adjacent-budget comparisons using the procedure in section S. Increasing the budget from $H = 40$ to $H = 80$ improves mean accuracy for all five model pairs, with statistically significant paired comparisons for four pairs. Increasing the budget from $H = 80$ to $H = 160$ gives smaller mean accuracy gains for all pairs, with statistically significant comparisons for three. Latency increases are statistically significant for every model pair in both comparisons.

Mean accuracy remains unchanged in 20 of the 65 pair–benchmark settings when increasing the budget from $H = 80$ to $H = 160$. These results show smaller average accuracy gains at the larger budget, while latency continues to increase.