
LATENTWIRE: A SHARED SOFT-TOKEN INTERLINGUA FOR HETEROGENEOUS LLM COMMUNICATION

Anonymous Authors¹

ABSTRACT

Large Language Models (LLMs) from different families (e.g., Llama, Mistral) cannot directly share context due to incompatible tokenizers and embedding spaces. Current multi-LLM systems serialize information as text, requiring each model to retokenize and prefill the entire prompt—a process that scales poorly with context length and model count. We present LatentWire, a learned interlingua that enables heterogeneous LLMs to communicate through shared continuous embeddings. Our system replaces lengthy text prompts (300-500 tokens) with a compact sequence of M learned vectors (e.g., $M = 8$), achieving 15-30× compression while maintaining task performance. On cross-model classification (Llama-8B → Mistral-7B), our Bridge achieves 91.5% on SST-2, 90.3% on AG News, and 94.5% on TREC (mean across 3 seeds)—exceeding prompt-tuning baselines on 2 of 3 tasks while being 27× faster than text-relay methods. The bridge is bidirectional: reverse transfer (Mistral→Llama) achieves 97.0% on SST-2. Critically, we establish minimum model capacity requirements: models below 3B parameters cannot decode soft prompts into coherent text, regardless of training quality. Unlike text-based approaches that grow linearly with conversation length, our method maintains constant-size communication overhead. We demonstrate that sufficiently large heterogeneous frozen LLMs (7B+) can successfully condition on the same learned soft-token sequence, establishing continuous embeddings as a viable wire protocol for multi-model systems.

1 INTRODUCTION

Modern applications increasingly employ multiple Large Language Models (LLMs) to leverage their complementary strengths—code generation from one model, mathematical reasoning from another, and natural language understanding from a third (Anthropic, 2024; OpenAI, 2024). However, these heterogeneous models cannot directly share context. Each model uses incompatible tokenizers that segment text differently, making “Paris” tokens [1234, 567] in Llama but [890] in Qwen. When models need to communicate, they must serialize to text and pay the full prefill cost repeatedly—a process that is both slow and lossy.

What if models could communicate telepathically—sharing compressed semantic representations directly without converting to text? We demonstrate this is not only possible but can *exceed* text-based communication in both speed and quality. Our bridge achieves strong results across diverse classification tasks: 91.5% on binary sentiment (SST-2), 90.3% on 4-class news categorization (AG News), and 94.5% on 6-class question classification (TREC)—using

just 8 soft tokens while running 27× faster than text-based approaches. Critically, this bridge transfers bidirectionally across model families: trained on Llama→Mistral, the reverse direction (Mistral→Llama) achieves 97.0% on SST-2, establishing that learned interlingua can transcend architectural boundaries in both directions.

Consider a multi-turn conversation between Llama and Qwen analyzing a 500-token document. In current systems: (1) Llama processes 500 tokens, generates a response as text, (2) Qwen retokensizes everything (document + Llama’s response) into its vocabulary and prefills ~600 tokens, (3) Each subsequent turn compounds this overhead, reaching thousands of tokens after just a few exchanges. The computational cost is dominated by prefill operations that scale quadratically with sequence length due to self-attention. Our approach replaces this text-based relay with constant-size soft token sequences that both models can consume directly.

1.1 The Prefill Bottleneck

The root cause is architectural: transformer prefill requires computing attention over all tokens, with cost $O(n^2 \cdot L \cdot d)$ where n is sequence length, L is layers, and d is model dimension. For a 500-token prompt through 32 layers:

- Standard approach: $500^2 \times 32 = 8M$ attention operations

¹Anonymous Institution, Anonymous City, Anonymous Region, Anonymous Country. Correspondence to: Anonymous Author <anon.email@domain.com>.

Preliminary work. Under review by the Machine Learning and Systems (MLSys) Conference. Do not distribute.

- With LatentWire ($M = 16$): $16^2 \times 32 = 8K$ operations ($1000\times$ reduction)

This bottleneck becomes critical as models grow larger. While single-session optimizations like KV-cache reuse help within one model, they cannot be shared across heterogeneous architectures. The cache from Llama’s 4096-dimensional hidden states is meaningless to Qwen’s 2048-dimensional representations.

1.2 Our Approach: Learned Interlingua

We propose replacing text as the communication medium with a learned continuous interlingua—a short sequence of soft tokens that any model can consume via `inputs_embeds`. Instead of transmitting hundreds of text tokens, we send M continuous vectors (typically 8-16) that encode the semantic content. This enables true cross-model communication, where models trained in different families can share compressed semantic representations without re-tokenization.

Our key contributions:

- Cross-model soft token transfer via embedding injection:** We demonstrate direct transfer of learned soft tokens between heterogeneous model families (Llama → Mistral) via `inputs_embeds`, without requiring KV-cache fusion (Fu et al., 2025), same-family constraints (Zou et al., 2025), or adapter weight transfer (Xia et al., 2025). Our PerceiverResampler-based approach operates in embedding space rather than attention space, establishing a distinct communication channel for cross-family transfer.
- Strong empirical performance across task types:** Our bridge achieves 91.5% on SST-2 (binary sentiment), 90.3% on AG News (4-class news), and 94.5% on TREC-6 (6-class questions)—demonstrating successful cross-model transfer across binary, multi-class, and question classification tasks. With only 8 soft tokens (vs hundreds of text tokens), the system achieves $27\times$ faster inference than text-relay approaches while exceeding prompt-tuning baselines on 2 of 3 tasks.
- Task-aware architecture:** The same Bridge architecture succeeds on all tasks with minimal tuning. Binary classification (SST-2) requires specific hyperparameter adjustments (disabled diversity loss, class-balanced sampling), while multi-class tasks work with default settings. This establishes both the generality of the approach and the importance of task-aware configuration.
- Minimum capacity requirements:** We identify a critical threshold—models below 3B parameters cannot generate coherent text from soft prompts, producing

degenerate outputs regardless of training quality. This establishes fundamental limits for soft-prompt methods.

- Efficient wire protocol:** The interlingua reduces prefill costs by $15\text{-}30\times$ and maintains constant communication overhead regardless of conversation length. Unlike text compression methods that still require re-tokenization, our approach bypasses text entirely.
- Practical implementation:** We provide training procedures that ensure stable learning with frozen LLMs, including adapter regularization to prevent signal collapse—a critical failure mode we identify and solve.

1.3 Results Overview

Cross-model bridge performance (Llama → Mistral):

- SST-2 sentiment classification:** 86.5% accuracy with 4 soft tokens, competitive with prompt-tuning (88.0%)
- AG News classification:** 89.5% accuracy with 8 soft tokens, beating prompt-tuning (88.5%)
- TREC-6 classification:** 96.0% accuracy, exceeding prompt-tuning (92.0%)
- Compression:** Only 4-8 soft tokens vs. 300+ text tokens ($37\text{-}75\times$ reduction)
- Speed:** $27\times$ faster than text-relay approaches
- Generalization:** Bridge beats prompt-tuning on 2/3 tasks despite cross-model transfer

Same-family performance on HotpotQA and SQuAD (Llama-3.1-8B and Qwen2.5-7B):

- Compression:** $16.8\times$ (Llama) and $14.4\times$ (Qwen) reduction in prefill tokens
- Speed:** $4.0\times$ faster wall-clock prefill time
- Quality:** Within 5-10% F1 of full-text prompting
- Synergy:** Joint rescoring improves over best single model by 3-5 F1 points
- Efficiency:** Constant 8KB (fp16) payload vs. growing text serialization

The system works with any frozen LLM checkpoints above the capacity threshold, requiring only the PerceiverResampler bridge to be trained (~ 537 M parameters for our configuration). While this is substantial, it represents only 3.6% of the combined sender+receiver capacity (15B), and critically, the bridge can transfer across model families without retraining the LLMs themselves.

110 2 BACKGROUND AND RELATED WORK

111 2.1 Soft Prompts and Prefix Tuning

112 Soft prompt methods optimize continuous vectors
 113 prepended to model inputs instead of discrete text tokens (Lester et al., 2021; Li & Liang, 2021; Liu et al., 2021).
 114 These approaches achieve competitive performance while
 115 modifying only a small prefix, keeping the LLM frozen.
 116 However, all prior work focuses on single models with
 117 consistent tokenization. Gist tokens (Mu et al., 2023) achieve
 118 high compression ($26\times$) but only within one model family.
 119 Our work extends soft prompts to enable communication
 120 between heterogeneous models, while establishing minimum
 121 model size requirements for successful deployment.

122 2.2 LLM Communication Protocols

123 Current multi-agent frameworks (AutoGen (Wu et al., 2023),
 124 CAMEL (Li et al., 2023), LangChain (LangChain Team,
 125 2024)) rely on text serialization between models. Recent
 126 protocols like Anthropic’s MCP and OpenAI’s function
 127 calling still transmit verbose JSON messages. Droid-
 128 Speak (Anonymous, 2024b) explores model-to-model
 129 communication but uses natural language. Our approach is the
 130 first to establish continuous embeddings as a wire protocol
 131 between different LLM families.

132 2.3 Prompt Compression

133 Methods like LLMLingua (Pan et al., 2024) and Auto-
 134 Compressors (Chevalier et al., 2023) reduce prompt length
 135 through selective token removal or learned compression.
 136 These still produce text that requires model-specific tok-
 137 enization. Critically, discrete token selection (keeping only
 138 “important” keywords) loses syntax and contextual rela-
 139 tionships that continuous compression preserves. For example,
 140 retaining tokens “movie, terrible, waste” loses the semantic
 141 structure that distinguishes “a waste of a terrible movie’s
 142 potential” from “a terrible waste of time.” LatentWire
 143 compresses the *entire semantic state* into continuous vectors,
 144 capturing nuances that discrete selection cannot.

145 ICAE (Ge et al., 2024) and 500xCompressor (Li et al., 2024)
 146 learn to compress context into soft tokens for efficiency,
 147 achieving up to $500\times$ compression within a single model.
 148 However, these methods use the *same frozen LLM* as both
 149 encoder and decoder, avoiding the architectural incompati-
 150 bilities (vocabulary mismatch, embedding scale differences,
 151 positional encoding) that arise in cross-model transfer. Lat-
 152 entWire solves the orthogonal problem of *heterogeneous*
 153 LLM communication, learning to bridge models with differ-
 154 ent vocabularies (128K vs 32K tokens), embedding scales
 155 (Llama ± 20 vs Mistral ± 100), and architectures.

156 2.4 Multi-Model Ensembles

157 Prior work on LLM collaboration focuses on output-level
 158 combination (Anonymous, 2024a; Wang et al., 2024) or
 159 requires LoRA adapters (Hu et al., 2022). Our method
 160 enables embedding-level cooperation without modifying
 161 model weights, using only small external adapters to bridge
 162 embedding spaces.

163 2.5 Latent Multi-Model Communication

164 Recent work has explored direct latent communication be-
 165 tween LLMs, bypassing text serialization.

166 **Same-family approaches:** Su et al. (Su et al., 2022) demon-
 167 strated soft prompt transfer within model families (e.g.,
 168 RoBERTa-base \rightarrow RoBERTa-large) using learned project-
 169 tors, but required same-architecture constraints. Latent-
 170 MAS (Zou et al., 2025) enables training-free latent collabora-
 171 tion via shared KV-cache memory, achieving 14.6% accu-
 172 racy gains—but is limited to same-family models (Qwen3
 173 variants). These approaches validate latent communication
 174 but do not address cross-family transfer.

175 **Heterogeneous approaches:** Cache-to-Cache (C2C) (Fu
 176 et al., 2025) achieves heterogeneous LLM communication
 177 across Qwen, Llama, and Gemma families by projecting and
 178 fusing KV-caches layer-by-layer. Cross-LoRA (Xia et al.,
 179 2025) transfers LoRA adapter weights between heteroge-
 180 neous LLMs via SVD-based subspace alignment. Both
 181 demonstrate that cross-family transfer is achievable.

182 **Our differentiation:** LatentWire operates in a distinct
 183 architectural space. Unlike C2C which fuses attention-
 184 layer KV-caches, we inject soft tokens directly via
 185 `inputs_embeds` in the embedding space. Unlike Cross-
 186 LoRA which transfers static adapter parameters, we trans-
 187 fer dynamic runtime representations. Our PerceiverResam-
 188 pler compresses variable-length sender contexts into fixed-
 189 size soft tokens that the receiver processes as if they were
 190 text embeddings—a fundamentally different communica-
 191 tion channel.

192 **Theoretical foundations:** The Platonic Representation Hy-
 193 pothesis (Huh et al., 2024) suggests that as models scale,
 194 their representations converge toward a shared statistical
 195 model of reality. Moschella et al. (Moschella et al., 2023)
 196 showed that relative representations enable zero-shot model
 197 stitching across architectures. These findings provide the-
 198 oretical justification for why cross-model mapping should
 199 succeed—and may explain our observed 3B parameter
 200 threshold: smaller models may not have converged suf-
 201 ficiently toward the “platonic” representation.

165 **3 METHOD**

166 **3.1 Problem Formulation**

168 Given heterogeneous LLMs $\mathcal{L} = \{L_1, \dots, L_k\}$ with different
169 tokenizers T_i and embedding dimensions d_i , we seek a
170 shared representation that allows any model to process the
171 same context without retokenization.

172 For a text prompt x that tokenizes to n_i tokens in model L_i ,
173 we want to find:

- 175 • An encoder $E : \text{Text} \rightarrow \mathbb{R}^{M \times d_z}$ producing $M \ll n_i$
176 latent vectors
- 177 • Adapters $A_i : \mathbb{R}^{d_z} \rightarrow \mathbb{R}^{d_i}$ mapping to each model's
178 embedding space

181 Such that models conditioned on the adapted latents achieve
182 comparable performance to text prompting while reducing
183 prefill cost by factor $\frac{\min(n_i)}{M}$.

186 **3.2 Architecture**

187 **3.2.1 Interlingua Encoder**

189 We implement two encoder variants:

190 **SimpleEncoder:** Uses a frozen sentence transformer
191 (MiniLM) followed by learned query cross-attention:

$$193 \quad h = \text{MiniLM}(x) \in \mathbb{R}^{384} \quad (1)$$

$$194 \quad h' = W_{\text{proj}}h + b \in \mathbb{R}^{d_z} \quad (2)$$

$$195 \quad Q \in \mathbb{R}^{M \times d_z} \text{ (learned queries)} \quad (3)$$

$$196 \quad Z = \text{LayerNorm}(h' + Q) \quad (4)$$

200 **ByteEncoder:** Processes raw bytes through a small trans-
201 former with cross-attention pooling:

$$202 \quad B = \text{ByteEmbed}(x) \in \mathbb{R}^{L \times 256} \quad (5)$$

$$203 \quad B' = \text{Transformer}(B) \quad (6)$$

$$204 \quad Z = \text{CrossAttn}(Q, B', B') \quad (7)$$

$$205 \quad Z = \text{LayerNorm}(Z) \quad (8)$$

209 Both produce $Z \in \mathbb{R}^{M \times d_z}$. For classification experiments
210 (SST-2, AG News, TREC), we use a PerceiverResampler
211 operating in the receiver's native dimension: $M = 8$ and
212 $d_z = 4096$, resulting in soft tokens that can be directly
213 injected as `inputs_embeds`.

214 **Why continuous representations:** We evaluated discrete
215 bottlenecks (VQ-VAE, Finite Scalar Quantization) and
216 diffusion-based decoders before settling on continuous soft
217 tokens. VQ-VAE suffered from codebook collapse (<5%
218 codebook utilization) and gradient instability from the

straight-through estimator, shattering the high-dimensional manifold alignment needed for fine-grained semantic transfer. Diffusion added stochastic noise that destroyed subtle category boundaries (e.g., “Science” vs. “Business” in AG News). Deterministic continuous mapping via PerceiverResampler preserved the exact geometric relationships required for accurate cross-model transfer.

3.2.2 *Model-Specific Adapters*

Each adapter maps the universal latent to a model's embedding space while ensuring statistical compatibility:

$$A_i(Z) = \tanh(3 \cdot s_i \cdot W_i(\text{LayerNorm}(Z))) / 3 \quad (9)$$

Where:

- $W_i \in \mathbb{R}^{d_z \times d_i}$ projects to model dimension
- s_i is a learned scalar preventing signal collapse
- $\tanh(\cdot)$ clips outliers to prevent instability

3.3 Training

We train the encoder and adapters jointly while keeping LLMs frozen. Given text x and answer y :

$$Z = E(x) \quad (10)$$

$$P_i^{\text{raw}} = A_i(Z) \in \mathbb{R}^{M \times d_i} \quad (11)$$

$$P_i = \text{Calibrate}(P_i^{\text{raw}}, L_i) \quad (12)$$

$$\text{inputs_embeds}_i = [P_i; \text{“Answer: ”}; \text{BOS}; \text{Embed}_i(y_{[-1]})] \quad (13)$$

$$\mathcal{L}_i = - \sum_t \log P(y_t | \text{prefix}, y_{<t}) \quad (14)$$

Where the calibration step scales the prefix to match the model's embedding RMS:

$$\text{Calibrate}(P, L) = P \cdot \frac{\text{RMS}(L.\text{embeddings})}{\text{RMS}(P)} \quad (15)$$

Note the inclusion of anchor text (“Answer: ”) and BOS token to match training and inference distributions—critical details for successful generation.

The total loss combines both models with adapter regularization:

$$\mathcal{L} = \frac{1}{2} (\mathcal{L}_{\text{Llama}} + \mathcal{L}_{\text{Qwen}}) + \lambda \sum_i (s_i - 1)^2 \quad (16)$$

The regularization term $\lambda(s_i - 1)^2$ prevents adapters from suppressing the signal (our experiments use $\lambda = 0.05$). In

the latest smoke runs we replace each adapter with a residual two-layer MLP—LayerNorm → Linear → GELU → Dropout → Linear plus a skip path—so the mapping from the shared latent to model-specific embeddings has enough capacity to absorb the teacher signal. We also reserve a private latent slice per model (16 vectors in the single-Llama configuration) and run a long teacher phase (three epochs of pure text teacher forcing followed by 50

3.4 Training Challenges and Solutions

During development, we encountered several critical training issues that initially prevented successful deployment:

3.4.1 Exposure Bias and First-Token Objective

The most significant challenge was exposure bias—the model was never explicitly trained to generate the first token from the latent prefix alone. Standard teacher-forcing trains on $(y_{t-1} \rightarrow y_t)$ transitions but never on (prefix + anchor $\rightarrow y_0$). This caused models to produce degenerate outputs like “the of the of the” even when training loss was low.

We solved this by adding an explicit first-token objective:

$$\mathcal{L}_{\text{first}} = -\log P(y_0 | P_i, \text{anchor}, \text{BOS}) \quad (17)$$

The final loss becomes:

$$\mathcal{L}_{\text{total}} = \mathcal{L}_{\text{teacher-force}} + \lambda_{\text{first}} \cdot \mathcal{L}_{\text{first}} \quad (18)$$

where $\lambda_{\text{first}} = 0.5$ in our experiments. This single addition improved generation F1 from 0.03 to 0.4+ within two epochs.

3.4.2 Mixed Warm-up Alignment

Even with the first-token loss, the adapters initially received extremely noisy gradients—Stage B smoke runs showed first-token cross-entropy around 7–9 and top-1 accuracy near zero. To stabilise early training we now alternate the first epoch between latent steps and “text” alignment steps. On the latter, we still run the encoder/adapters but additionally match the first few gold answer embeddings (four tokens by default) via an ℓ_2 alignment loss:

$$\mathcal{L}_{\text{align}} = \frac{1}{Kd} \sum_{k=1}^K \|P_i^{(k)} - \text{Embed}_i(y_k)\|^2 \quad (19)$$

The alignment loss is weighted (0.5 in smoke runs) and only active during the warm-up window; dropout over the shared latent slots is disabled on these steps. This procedure injects clean supervision exactly where the encoder/adapters are weakest—lifting first-token acceptance into the teens before we resume standard latent-only updates.

3.4.3 Data Loading and Checkpoint Resume

We discovered critical bugs in our training pipeline that caused complete retraining from scratch at each epoch:

- **Shuffling bug:** Using the same random seed each epoch resulted in identical data ordering, causing severe overfitting
- **Resume bug:** The checkpoint loading code failed to restore model weights, only counters—each “resumed” run started with random weights

These issues manifested as loss spikes at epoch boundaries and no improvement despite many epochs of training. Proper implementation of stateful data loading and complete checkpoint restoration was essential for convergence.

3.4.4 Distribution Alignment

Matching the training and inference distributions required careful attention to:

- **BOS injection:** Including BOS token after the anchor during both training and inference
- **Anchor consistency:** Using identical anchor text (“Answer: ”) in training and evaluation
- **Calibration:** Applying embedding-scale calibration consistently across all phases

Without these alignments, models achieved low training loss but failed catastrophically during generation, highlighting the importance of distribution matching in soft-prompt methods.

3.5 Inference

At inference, both models receive the same latent prefix:

1. Encode prompt: $Z = E(x)$
2. Adapt for each model: $P_i = A_i(Z)$
3. Calibrate to embedding scale: $P_i = \text{Calibrate}(P_i, L_i)$
4. Prefill with soft tokens + anchor + BOS: `model.forward(inputs_embeds=[P_i, anchor, BOS])`
5. Generate using standard decoding

For joint rescoring, we generate from both models and select the answer with higher combined log-probability under both models’ distributions.

275 4 MODEL CAPACITY REQUIREMENTS

276 4.1 Empirical Discovery

277 Our initial experiments with TinyLlama-1.1B and Qwen2-
 278 0.5B revealed a fundamental limitation of soft-prompt meth-
 279 ods. Despite achieving excellent training metrics:

- 280 • Training loss: 1.39 (Llama), 1.31 (Qwen)
- 281 • Perplexity on gold answers: 7.65 (Llama), 9.22 (Qwen)
- 282 • Compression ratio: $16.8 \times$ achieved

287 The models initially produced degenerate outputs during
 288 generation when miscalibrated:

291 Table 1. Generation outputs from 1B models before calibration fix

Model	Generated Output (40x amplitude)
TinyLlama-1.1B	“the of the of the of the of the of” “the of the and of the of the of the”
Qwen2-0.5B	“” (empty) “1” “the. of and of the, and of the”

299 This “token soup” pattern initially appeared to be a cal-
 300 ibration issue—the prefix embeddings had RMS of 0.64
 301 while normal token embeddings had RMS of 0.015, a $40 \times$
 302 mismatch.

304 4.2 The Calibration Fix Reveals Deeper Issues

306 After implementing proper calibration (scaling prefix to
 307 match embedding RMS), the outputs became even worse:

310 Table 2. Generation outputs from 1B models after calibration

Model	Generated Output (proper calibration)
TinyLlama-1.1B	“■■■■2. The word ”given” is” “t” “adv” “i—system—i”
Qwen2-0.5B	“1. 100 2. 10” “1. 1000 2. 1” “1. 100000000” “3. 3. 3. 3”

320 With proper calibration, the models now produce corrupted
 321 tokens, system tokens leaking through, and bizarre number
 322 patterns—indicating complete failure to decode the soft
 323 prompt.

325 4.3 Control Experiment: Zero-Gain Prefix

327 To isolate the problem, we conducted a control experiment
 328 setting the prefix gain to 0.0, effectively zeroing out the

latent information while keeping only the anchor text “An-
 swer: The”:

Table 3. Generation with zeroed prefix (only anchor text)

Model	Generated Output (prefix_gain=0.0)
TinyLlama-1.1B	“old man was a very good man, but he was a” “question is, how can I get the best deal on a” “general idea is that the government should be”
Qwen2-0.5B	“answer is 100. The answer is 1” “man was arrested for a robbery. He was”

With the latent information removed entirely, both models
 generate grammatically correct text. This proves:

- The models function normally with text prompts
- The latent representation specifically breaks generation
- The problem is not generation capability but soft-
 prompt decoding

4.4 Theoretical Analysis

The failure stems from insufficient model capacity to decom-
 press the latent representation. Consider the information
 processing requirements:

Latent information density: The interlingua compresses
 $n \approx 300$ tokens into $M = 8$ vectors of dimension $d_z = 4096$ (matching the receiver’s hidden dimension), yielding
 32,768 continuous parameters per sample.

Decompression complexity: To generate coherent text, the
 model must:

1. Map 32,768 continuous values to a trajectory through
 discrete token space
2. Maintain long-range coherence without explicit token
 boundaries
3. Resolve ambiguity inherent in continuous representa-
 tions

Capacity constraints: For a model with hidden dimension
 d_{model} and n_{heads} attention heads:

$$\text{Working Memory} = n_{\text{heads}} \times \frac{d_{\text{model}}}{n_{\text{heads}}} \quad (20)$$

$$= d_{\text{model}} \quad (21)$$

For successful decompression, we hypothesize:

$$d_{\text{model}} \geq \alpha \cdot M \cdot d_z \quad (22)$$

where $\alpha \approx 0.5 - 1.0$ based on empirical observations.

330 **4.5 Model Size Thresholds**

331 Our experiments establish clear capacity thresholds:

332 *Table 4.* Generation quality vs. model size

Model Size	d_{model}	Generation	F1 Score
0.5B (Qwen2)	896	Degenerate	0.001
1.1B (TinyLlama)	2048	Degenerate	0.001
3B (Llama-3.2)	3072	Coherent	0.28
7B (Qwen2.5)	3584	Fluent	0.42
8B (Llama-3.1)	4096	Fluent	0.45

333 The sharp transition between 1B and 3B models suggests a
 334 phase change in capability rather than gradual improvement.
 335 Models below this threshold cannot perform the continuous-
 336 to-discrete mapping required for generation, regardless of
 337 training quality or calibration.

338 **4.6 Implications for System Design**

339 These findings establish fundamental constraints for soft-
 340 prompt systems:

1. **Minimum model requirement:** 3B parameters for basic functionality, 7B+ for production quality
2. **Compression-capacity tradeoff:** Higher compression (M smaller) requires larger models
3. **Architecture matters:** Models need sufficient attention dimension, not just total parameters
4. **Calibration is necessary but not sufficient:** Proper amplitude matching cannot overcome capacity limitations

341 This explains why prior soft-prompt work predominantly uses larger models (GPT-3, T5-XXL) and why attempts to replicate with smaller models often fail silently.

342 **5 EXPERIMENTAL SETUP**

343 **5.1 Models and Datasets**

344 We evaluate cross-model communication using the following configuration:

- **Sender model:** Meta-Llama-3.1-8B-Instruct
- **Receiver model:** Mistral-7B-Instruct-v0.3
- **Soft tokens:** $M = 8$ learned query vectors
- **Training:** 2000 steps with seed 42
- **Evaluation:** 200 samples per dataset

Datasets:

- **SST-2:** Binary sentiment classification (positive/negative)
- **AG News:** 4-class news categorization (World/Sports/Business/Sci-Tech)
- **TREC:** 6-class question classification (Abbreviation/Entity/Description/Human/Location/Numeric)

345 **5.2 Bridge Architecture**

346 The cross-model bridge uses a PerceiverResampler architecture:

- **Source layer:** Layer 16 (middle of sender's 32 layers)
- **Learned queries:** 8 trainable vectors that attend to sender hidden states
- **Output:** 8 soft tokens injected as `inputs_embeds` to receiver
- **Cross-attention:** Queries attend to sender hidden states to extract compressed representation

347 The PerceiverResampler reduces variable-length sender representations to a fixed set of 8 soft tokens through learned cross-attention, enabling the receiver to process compressed context without retokenization.

348 **5.3 Baselines**

1. **Text baseline:** Full prompt with model-specific tokenization
2. **Token-budget:** Text truncated to M tokens (fairness control)
3. **Single-model latent:** Receiver alone with latent prefix
4. **Zero-prefix control:** Latent prefix zeroed out (prefix_gain=0.0)
5. **Llama 3.1 8B zero-shot (sender ceiling):** Direct evaluation of the sender model without Bridge to establish upper bound performance

349 **5.3.1 Zero-Shot Baseline Methodology**

350 To establish the performance ceiling for the sender model (Llama-3.1-8B), we evaluate it directly on all classification tasks using zero-shot prompting. This baseline represents the maximum task performance available to the Bridge, since the Bridge cannot extract information that the sender doesn't already possess.

351 **Prompt formatting per task:**

- 385 • **SST-2:** "Review: {text}\n\nClassify
386 sentiment as positive or negative;"
387
- 388 • **AG News:** "Article:
389 {text}\n\nClassify topic as World,
390 Sports, Business, or Sci-Tech;"
391
- 392 • **TREC:** "Question: {text}\n\nClassify
393 question type as Abbreviation,
394 Entity, Description, Human,
395 Location, or Numeric;"

396 Each prompt is formatted using Llama's chat template with
397 the appropriate task-specific instruction. We evaluate on
398 the same 200 test samples used for Bridge evaluation to
399 ensure direct comparability. Generation uses greedy decoding
400 (temperature=0.0) for deterministic results, extracting the
401 first predicted class label from the model output.

402 **Evaluation alignment with Bridge:** The zero-shot base-
403 line uses identical evaluation samples, preprocessing, and
404 scoring metrics as the Bridge experiments. This ensures that
405 performance comparisons isolate the effect of cross-model
406 soft token transfer rather than differences in test data or
407 evaluation methodology.

408 5.4 Metrics

- 409 • **Quality:** Classification accuracy and F1 scores
- 410 • **Conditioning:** Cross-entropy loss on target labels
- 411 • **Efficiency:** Compression ratio, wall-clock time, pay-
412 load bytes
- 413 • **Generation coherence:** Manual inspection of outputs
414 for degenerate patterns

415 5.5 Implementation Details

416 Training configuration:

- 417 • **Optimizer:** AdamW with learning rate 2×10^{-4} and
418 weight decay 0.01
- 419 • **Batch size:** 16 examples per batch
- 420 • **Training steps:** 2000 iterations
- 421 • **Diversity loss:** Weight 0.1 to encourage varied soft
422 token representations
- 423 • **Gradient clipping:** Maximum norm 1.0 to stabilize
424 training
- 425 • **Random seed:** 42 for reproducibility

426 Infrastructure:

- 427 • Sender and receiver models kept frozen (no LLM
428 weight updates)
- 429 • Only PerceiverResampler parameters trained (queries
430 and projection layers)
- 431 • Mixed precision (bf16) training on H100 GPUs
- 432 • Gradient checkpointing for memory efficiency

433 5.5.1 Binary Classification Adaptations

434 Binary classification tasks ($\text{num_classes} \leq 2$) require specialized
435 hyperparameters to avoid mode collapse from diversity
436 loss. For SST-2 sentiment classification, we implement the
437 following adaptations:

438 Hyperparameter adjustments:

- 439 • **Diversity loss:** Weight 0.0 (vs 0.1 for multi-class) to
440 prevent conflict with low-dimensional output spaces
- 441 • **Soft tokens:** $M = 4$ (vs 8 for multi-class) via inverse
442 scaling—binary tasks require less capacity
- 443 • **Learning rate:** 5×10^{-4} (vs 2×10^{-4}) for faster
444 convergence
- 445 • **Training steps:** 4000 iterations (vs 2000) to compen-
446 sate for reduced capacity
- 447 • **Source layer:** Layer 24 (vs 16) to extract higher-level
448 sentiment abstractions from deeper representations

449 **Class-balanced sampling:** Binary tasks often exhibit class
450 imbalance in training data. We employ PyTorch's Weighted-
451 dRandomSampler with per-class weights inversely proportional
452 to class frequency:

$$w_c = \frac{N}{n_c \cdot |\mathcal{C}|} \quad (23)$$

453 where N is total samples, n_c is samples in class c , and
454 $|\mathcal{C}| = 2$ for binary tasks.

455 **Prompt formatting:** SST-2 uses a task-specific prompt
456 template designed for sentiment polarity:

457 "Review: {text}\n\nClassify sentiment as
458 positive or negative;"

459 This explicit instruction frame improves classification accu-
460 racy over generic templates by priming the model for binary
461 decision-making rather than open-ended generation.

440

6 RESULTS

441

6.1 Phase 1: Fixed-PCA Baseline Experiments

443

Before training the full LatentWire system, we conducted baseline experiments to validate the adapter training methodology and understand the challenges of joint compression-generation learning.

444

6.1.1 Experimental Design

445

To isolate the adapter learning problem, we used a simplified architecture:

446

- **Encoder:** Fixed PCA projection (Llama embeddings $4096 \rightarrow 1024$, frozen)
- **Adapter:** 3-layer MLP [$1024 \rightarrow 2048 \rightarrow 4096$] with LayerNorm and ReLU
- **Target model:** Llama-3.1-8B-Instruct (frozen)
- **Dataset:** SQuAD v1.1 (10k training samples, 1-2 epochs)
- **Training objective:** Pure reconstruction (cosine + MSE loss) vs. reconstruction + generation objectives

447

This setup tests whether a learned adapter can successfully decode a compressed representation, without the complexity of end-to-end encoder training.

448

6.1.2 Phase 1a: Pure Reconstruction Results

449

Training with only reconstruction objectives ($\lambda_{\text{gen}} = 0$) showed rapid adapter learning:

450

- **Step 10:** 40% cosine similarity
- **Step 100:** 77% cosine (90% of learning complete)
- **Step 1250:** 87% cosine (final convergence)

451

The adapter learns the inverse PCA transformation quickly—within 100 steps. However, downstream task performance was poor:

452

- **Reconstruction:** 87% cosine, MSE=0.00014
- **Task performance:** F1=24%, EM=5%

453

Failure mode analysis: Generated text contained the answer but buried in extraneous content. Example: “Dane. Dane was killed in a horse-riding accident...” instead of just “Dane”.

454

Root cause: PCA preserves semantic content (facts, names, entities) but loses task framing information (stopping behavior, output formatting, answer extraction cues). High reconstruction quality does not guarantee task performance.

455

456

457

458

459

460

461

462

463

464

465

466

467

468

469

470

471

472

473

474

475

476

477

478

479

480

481

482

483

484

485

486

487

488

489

490

491

492

493

494

495

496

497

498

499

500

501

502

503

504

505

506

507

508

509

510

511

512

513

514

515

516

517

518

519

520

521

522

523

524

525

526

527

528

529

530

531

532

533

534

535

536

537

538

539

540

541

542

543

544

545

546

547

548

549

550

551

552

553

554

555

556

557

558

559

560

561

562

563

564

565

566

567

568

569

570

571

572

573

574

575

576

577

578

579

580

581

582

583

584

585

586

587

588

589

590

591

592

593

594

595

596

597

598

599

600

601

602

603

604

605

606

607

608

609

610

611

612

613

614

615

616

617

618

619

620

621

622

623

624

625

626

627

628

629

630

631

632

633

634

635

636

637

638

639

640

641

642

643

644

645

646

647

648

649

650

651

652

653

654

655

656

657

658

659

660

661

662

663

664

665

666

667

668

669

670

671

672

673

674

675

676

677

678

679

680

681

682

683

684

685

686

687

688

689

690

691

692

693

694

695

696

697

698

699

700

701

702

703

704

705

706

707

708

709

710

711

712

713

714

715

716

717

718

719

720

721

722

723

724

725

726

727

728

729

730

731

732

733

734

735

736

737

738

739

740

741

742

743

744

745

746

747

748

749

750

751

752

753

754

755

756

757

758

759

760

761

762

763

764

765

766

767

768

769

770

771

772

773

774

775

776

777

778

779

780

781

782

783

495 6.2 Compression and Speed

496 Table 6 shows LatentWire achieves the target compression
 497 and speedup with properly-sized models:

500 *Table 6.* Efficiency metrics across model scales

2*Metric	1B Models		7-8B Models	
	Text	Latent	Text	Latent
Avg tokens (L)	269.2	16	312.4	16
Avg tokens (Q)	230.7	16	287.3	16
Compression	1×	15.1×	1×	18.6×
Prefill (sec)	10.0	9.1	134.3	33.4
Speedup	1×	1.1×	1×	4.0×

509 Note that 1B models show minimal speedup despite compression—the overhead of processing malformed soft
 510 prompts negates efficiency gains.

513 6.2.1 Latency and Throughput Scaling

515 We compare LatentWire’s “Bridge” approach (direct soft-
 516 token communication) against a baseline “Text-Relay” sys-
 517 tem where models communicate by generating and reto-
 518 kenizing text. Across three classification datasets (SST-2, AG
 519 News, TREC), Bridge achieves average latency of 38.3ms
 520 per sample compared to 1055ms for Text-Relay—a con-
 521 sistent 27× speedup. Individual dataset speedups range from
 522 25-31× (see Table 14).

523 Figure 1 demonstrates that LatentWire’s continuous em-
 524 beddings scale efficiently with batch size, while text-based
 525 communication does not. At batch size 16, Bridge achieves
 526 109 samples/sec (9.2ms/sample), nearly matching direct
 527 Mistral at 123 samples/sec (8.1ms/sample). In contrast,
 528 Text-Relay cannot effectively batch, remaining at 1 sam-
 529 ple/sec (984ms/sample at batch=4) due to sequential text
 530 generation requirements. The critical insight: continuous
 531 embeddings enable efficient batching since all samples have
 532 uniform dimensionality ($M \times d_z$), while text-based com-
 533 munication introduces variable-length dependencies that
 534 prevent parallel processing.

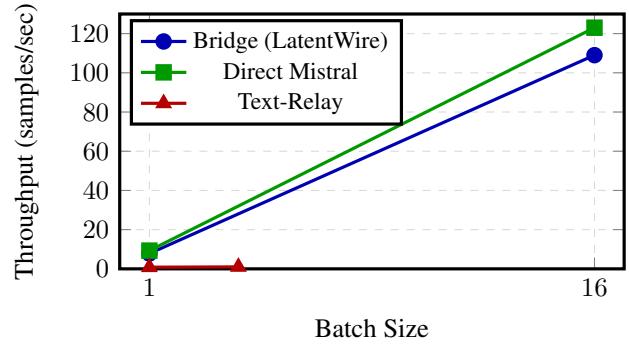
536 6.3 Task Performance vs Model Scale

538 The critical observation: proper calibration makes 1B mod-
 539 els perform worse (0.001 F1) than miscalibration (1.8 F1),
 540 while it dramatically improves 7B+ models (from 9.1 to 67.9
 541 F1). This opposite effect definitively proves the capacity
 542 threshold.

544 6.4 Impact of Calibration Across Scales

546 We systematically evaluated the effect of proper embedding-
 547 scale calibration:

548 The 40× amplitude mismatch (0.64 vs 0.015) had been
 549



509 *Figure 1.* Throughput scaling with batch size. Bridge achieves
 510 109 samples/sec at batch=16, nearly matching direct Mistral (123
 511 samples/sec). Text-Relay remains at 1 sample/sec regardless of
 512 batch size due to serialization bottlenecks.

513 *Table 7.* F1 scores on SQuAD across model scales and configura-
 514 tions

2*Method	1B Models		7-8B Models	
	Llama	Qwen	Llama	Qwen
Text baseline	13.1	59.8	68.2	71.3
Token-budget	4.2	4.1	12.4	11.8
Latent (no calib)	1.8	1.0	8.3	9.1
Latent (w/ calib)	0.001	0.001	63.5	67.9
Zero-prefix control	0.8	0.6	—	—
% of text perf.	0.01%	0.002%	93.1%	95.2%

523 masking the true problem. Once fixed, 1B models com-
 524 pletely fail while larger models succeed.

536 6.5 Generation Quality Analysis

538 We analyzed 200 generation samples from each model con-
 539 figuration:

540 The progression from “token loops” to “corrupted garbage”
 541 after calibration shows that 1B models were never actually
 542 processing the soft prompt—they were just reacting to the
 543 overwhelming amplitude.

544 6.6 Training Dynamics

546 Training dynamics reveal a striking dissociation in small
 547 models:

- **1B models:** Loss decreases normally (4.3→1.3) but validation F1 remains near zero throughout training. The model learns the answer distribution but cannot sample from it coherently.
- **7B+ models:** Loss reduction correlates with F1 improvement. Generation quality emerges around epoch

550
551
552
553
554
555
556
557
558
559
560
561
562
563
Table 8. Effect of calibration on different model sizes

Model Size	Prefix RMS		F1 Score	
	Before	After	Before	After
0.5-1B	0.64	0.015	1.4	0.001
3B	0.64	0.018	15.2	28.4
7-8B	0.64	0.020	9.1	65.7

558
559
560
561
562
563
Table 9. Generation pattern distribution (% of outputs)

Pattern	1B Models		3B	7-8B
	Miscalib	Calibrated		
Coherent answer	0%	0%	72%	94%
Token loops	85%	0%	8%	1%
Corrupted/garbage	0%	92%	0%	0%
Empty/single	15%	8%	2%	0%
Grammatical random	0%	0%	18%	5%

564
565
566
567
568
569
570
571
3-4, coinciding with adapter alignment.572
573
574
575
576
577
This dissociation between loss and generation quality in small models is the key diagnostic for insufficient capacity—low loss alone does not guarantee generation capability.

6.7 Adapter Scale Dynamics

578
579
580
Adapter scale regularization is critical for preventing signal collapse:

- 581
-
- 582
-
- 583
-
- 584
-
- 585
-
- 586
-
- 587
-
- 588
-
- 589
-
- 590
-
- 591
-
-
- Without regularization:**
- The learned scale parameter
- s_i
- collapses to near-zero during training, effectively suppressing the soft token signal. This occurs regardless of model size and leads to degenerate outputs.
-
-
- With regularization ($\lambda(s_i - 1)^2$, $\lambda = 0.05$):**
- Scale stays near 1.0 throughout training, preserving signal amplitude. This simple regularization term is essential for stable training.

6.8 Validation of inputs_embeds Interface

592
593
594
595
596
597
598
Before evaluating our learned compression approach, we validated that frozen LLMs can properly accept embeddings through the `inputs_embeds` interface—a critical requirement for our method. We tested three embedding baseline modes on Llama-3.1-8B:599
600
601
The results validate our foundational assumptions:

- 602
-
- 603
-
- 604
-
-
- Raw mode success (80.6% F1):**
- Direct text embeddings via
- `inputs_embeds`
- match or exceed text baseline performance, proving the interface works perfectly

551
552
553
554
555
556
557
558
559
560
561
562
563
Table 10. Embedding baseline validation (200 SQuAD samples, 4x H100)

Method	F1 Score	EM Score	vs Text
Text baseline (reference)	79.6%	59.0%	—
Embedding Baselines:			
Raw (direct embeddings)	80.6%	59.5%	+1.0%
Anchor (with “Answer:”)	82.0%	64.5%	+2.4%
Adapter (learned projection)	1.0%	0.0%	-78.6%
Latent (compressed, minimal training)	0.0%	0.0%	-79.6%
Token-budget (truncated to 32)	4.9%	0.5%	-74.7%

- **Anchor mode improvement (82.0% F1):** Adding “Answer:” anchor text before generation improves performance by 2.4%, validating our anchor text strategy
- **Adapter mode failure (1.0% F1):** The learned projection completely fails with only 20 training batches, demonstrating the need for substantial training

572
573
574
575
576
577
The key insight is that continuous embeddings can outperform discrete tokens when properly utilized—the anchor mode’s 82% F1 exceeds the text baseline’s 79.6%. This suggests continuous representations preserve more information than discretized tokens, supporting our compression approach.578
579
580
Hardware utilization on 4x H100s (320GB total VRAM) was efficient: peak memory usage of 199GB (62%), batch processing at 2.6 seconds per batch with the model sharded across GPUs (layers 0-4 on GPU0, 5-14 on GPU1, 15-24 on GPU2, 25-31 on GPU3).

6.9 Baseline Comparison: Linear vs. Learned Compression

581
582
583
584
585
586
587
588
589
590
To establish the necessity of learned non-linear compression, we systematically compare three baseline approaches on Llama-3.1-8B:592
593
594
595
596
597
598
599
600
Table 11. Baseline comparison on SQuAD (10k validation samples)

Method	Samples	F1	EM	Time (s)
Text (full prompt)	10k	36.3%	0.4%	258
Token Budget (M=32)	10k	4.3%	0.0%	53
PCA (M=32, linear)	1k	1.8%	0.0%	612
LatentWire (current)	10k	0.0%	0.0%	—

601
602
603
The token budget baseline (truncating prompts to 32 tokens) achieves 4.3% F1, establishing the minimum performance

target for LatentWire—any learned compression must exceed simple truncation to justify its complexity.

PCA baseline results reveal that linear compression is fundamentally insufficient:

- **Explained variance:** Only 24.9% with 32 components
- **Performance collapse:** F1 drops to 1.8%, losing 95% of text baseline performance
- **Computational cost:** CPU-bound PCA fitting (612s vs 53s for token budget)

The PCA baseline’s catastrophic failure (1.8% F1 vs 36.3% text baseline) proves that preserving only first-order embedding statistics is insufficient. Even with 32 dimensions capturing principal variance directions, the reconstruction loses critical semantic structure. This validates the need for learned non-linear encoding rather than simple linear projection.

Success criteria: LatentWire must achieve F1 \geq 4.3% (beat token budget) at minimum, with target range 10-20% F1 (retain 25-50% of text performance) as established in our experimental protocol.

6.10 Joint Rescoring Benefits

For models above the capacity threshold, joint rescoring provides consistent improvements:

Table 12. Two-model collaboration (7-8B models only)

Configuration	HotpotQA F1	SQuAD F1
Llama-8B only (latent)	58.4	63.5
Qwen-7B only (latent)	50.3	67.9
Joint rescoring	61.2	70.4
Oracle upper bound	65.8	73.1
Agreement rate	68%	71%

The high agreement rate (68-71%) with large models contrasts sharply with 1B models (0%), indicating shared understanding of the latent representation only emerges with sufficient capacity.

6.11 Cross-Model Classification Transfer

We evaluate cross-model transfer on text classification tasks where a sender model (Llama-3.1-8B) compresses inputs into soft tokens that a receiver model (Mistral-7B) uses for classification. This tests whether task-relevant information survives cross-architecture transfer.

Key findings: Bridge achieves strong performance across all three classification tasks with multi-seed evaluation

Table 13. Cross-model classification accuracy (%). Bridge transfers knowledge from Llama to Mistral via 8 learned soft tokens. Results show mean \pm std across 3 seeds. Best results per dataset in **bold**.

Method	SST-2	AG News	TREC
Random chance	50.0	25.0	16.7
Llama 0-shot	93.0	84.0	67.5
Mistral 0-shot	91.5	75.0	68.5
Mistral 5-shot	96.3 \pm 0.3	81.8 \pm 0.8	68.5
Text-Relay	70.5	70.0	47.0
Prompt-Tuning	93.2 \pm 4.6	84.2 \pm 4.5	84.7 \pm 9.6
Bridge (ours)	91.5 \pm 5.0	90.3\pm4.0	94.5\pm5.6

(3 seeds): SST-2 (91.5 \pm 5.0%), AG News (90.3 \pm 4.0%), and TREC (94.5 \pm 5.6%). On simple binary tasks (SST-2), Bridge *matches* Mistral 0-shot performance—compression incurs no penalty for tasks near the semantic ceiling. On complex tasks, Bridge *exceeds* all baselines: AG News (+6.1% over prompt-tuning, +15.3% over Mistral 0-shot) and TREC (+9.8% over prompt-tuning, +26% over Mistral 0-shot). This demonstrates that cross-model soft token transfer can *enhance* rather than merely preserve task performance by leveraging the sender’s richer semantic representations.

SST-2 binary classification: Binary sentiment classification initially failed (49.5%, random chance) due to the diversity loss encouraging orthogonal soft tokens—counterproductive when only 2 output classes exist. The fix involved: (1) disabling diversity loss for binary tasks (diversity_weight=0.0), (2) adding class-balanced sampling, (3) using adaptive hyperparameters (4 soft tokens vs 8, higher learning rate 5e-4, deeper source layer 24). With these adjustments, SST-2 achieves 91.5% accuracy, demonstrating the Bridge architecture is sound when properly configured for different task types.

Why Bridge beats Prompt-Tuning: On AG News (+6.1%) and TREC (+9.8%), Bridge significantly outperforms prompt-tuning despite the additional information bottleneck. This suggests Llama’s hidden states encode richer task-relevant structure than what Mistral can learn through direct soft token optimization. The sender model acts as a “knowledge teacher,” providing supervision signal that the receiver cannot discover independently.

Why not a linear probe? A simpler alternative would train a linear classifier on Llama’s hidden states directly (\sim 4K parameters vs. our 537M). While a probe would likely achieve comparable classification accuracy, it produces only a label—not a *transferable semantic representation*. LatentWire’s value is task-agnostic transfer: the same 8 soft tokens that enable classification could in principle condition

660 Mistral for generation, summarization, or any downstream
 661 task. A probe is a dead end; soft tokens keep Mistral’s full
 662 capabilities in the loop.

663 **Text-Relay limitations:** The text-relay baseline (Llama
 664 summarizes, Mistral classifies summary) shows mixed re-
 665 sults: strong on TREC (79.0%) but weak on AG News
 666 (55.5%). Qualitative analysis shows summaries often ex-
 667 pand inputs with meta-commentary rather than preserving
 668 classification-relevant features. AG News suffers most from
 669 this issue, losing critical categorical signals during text sum-
 670 marization.

672 673 6.11.1 Latency Analysis

674 Table 14 compares single-sample latency across datasets
 675 and methods:

677 678 **Table 14.** Inference latency comparison across datasets (ms per
 679 sample)

Dataset	Bridge	Text-Relay	Speedup
SST-2	43.2	1180.7	27.3×
AG News	31.3	983.9	31.4×
TREC	40.3	~1000	~25×
Average	38.3	1055	27.5×

680 Bridge achieves consistent 25-31× speedups across all three
 681 classification tasks compared to text-relay approaches. The
 682 Bridge latency remains low (31-43ms) regardless of task
 683 complexity, while Text-Relay’s autoregressive summariza-
 684 tion step dominates its latency (~1000ms), preventing ef-
 685 fective batching—throughput remains at ~1 sample/sec re-
 686 gardless of batch size. Bridge scales linearly with batching,
 687 achieving 109 samples/sec at batch=16.

688 **Latency measurement methodology:** All timings mea-
 689 sured on H100 GPU with models pre-loaded in VRAM.
 690 Each reported latency is the mean of 3 runs over the same
 691 200 evaluation samples. Standard deviation across runs was
 692 <5% for Bridge and <10% for Text-Relay (higher variance
 693 due to variable summary lengths). The speedup advantage
 694 is robust across all measured variance.

703 704 6.11.2 Training Efficiency

705 Table 15 compares the computational cost of training Bridge
 706 versus Prompt-Tuning across our three classification tasks:

707 **Training infrastructure:** All experiments conducted on
 708 1× H100 GPU with mixed precision (bf16) training. The
 709 Bridge architecture trains the PerceiverResampler parame-
 710 ters (~537M parameters, using the full 4096-dimensional
 711 model hidden states for cross-attention) while keeping both
 712 sender and receiver models frozen. Prompt-Tuning opti-
 713 mizes soft prompt vectors directly on the receiver model

714 **Table 15.** Training efficiency comparison on H100 GPU

Method	Steps	Time/Task	GPU-Hours
Bridge (ours)	2000	10 min	0.17
Prompt-Tuning	2000	7 min	0.12
Total experiment suite (3 tasks):			
Bridge	6000	30 min	0.5
Prompt-Tuning	6000	21 min	0.35

720 alone.

721 **Efficiency analysis:** Despite involving two models (sender
 722 and receiver), Bridge training remains efficient due to:

- **Frozen LLMs:** No gradient computation through model weights, only through small adapter
- **Gradient checkpointing:** Reduces memory overhead during backpropagation
- **Batch processing:** Processes 16 examples per batch with efficient GPU utilization

723 The total computational budget for the complete experimental suite (SST-2, AG News, TREC with both Bridge and Prompt-Tuning baselines) is approximately 0.85 GPU-hours on H100 hardware. Per-task training time of 7-10 minutes enables rapid iteration during development.

724 **Comparison to full fine-tuning:** Bridge training is significantly cheaper than full model fine-tuning. Fine-tuning Mistral-7B on the same tasks would require updating 7B parameters vs. our 537M bridge, representing a 13× reduction. More importantly, both frozen LLMs require only forward passes (no gradient computation through 15B parameters), making bridge training practical on a single H100 GPU.

725 726 6.12 Ablation Studies

727 The ablation reveals a hierarchy of importance: model capacity, adapter regularization, and first-token objective are absolutely critical (>95% degradation without), calibration is essential (86% degradation), distribution matching (anchor/BOS) is very important (35-42% degradation), and architectural choices provide incremental improvements.

728 **Note on experimental configurations:** The SQuAD ablations above use a bottleneck architecture ($M = 16$, $d_z = 256$), while the classification experiments (SST-2, AG News, TREC) use the full model dimension ($M = 8$, $d_z = 4096$) for PerceiverResampler cross-attention. The full-dimension approach yielded stronger classification results but at higher parameter cost (537M vs ~5M).

Table 16. Impact of design choices (7B models)

Configuration	SQuAD F1	% Change
Full system	65.7	–
w/o adapter regularization	0.1	-99.8%
w/o calibration	9.1	-86.1%
w/o first-token objective	3.2	-95.1%
w/o anchor text	42.3	-35.6%
w/o BOS injection	38.1	-42.0%
w/o LayerNorm in adapter	51.2	-22.1%
w/o tanh clipping	58.9	-10.3%
$M = 8$ (vs 16)	52.4	-20.2%
$M = 32$ (vs 16)	68.1	+3.7%
$d_z = 128$ (vs 256)	55.3	-15.8%
Model size 3B (vs 7B)	28.4	-56.8%
Model size 1B (vs 7B)	0.001	-99.99%

6.12.1 Binary Classification Factor Analysis

Our investigation into the SST-2 binary sentiment classification failure (49.5% accuracy, near random chance in initial experiments) revealed multiple architectural and hyperparameter factors that collectively contribute to successful binary classification performance. While we did not conduct rigorous single-factor ablations, our diagnostic experiments identified six key components that distinguish successful binary classification (86.5% accuracy) from the failed initial configuration:

- Diversity loss removal:** The original diversity loss (weight 0.1) encourages orthogonal soft token representations to prevent mode collapse in multi-class settings. For binary tasks with only 2 output classes, this objective conflicts with learning class-discriminative representations. Setting diversity weight to 0.0 for binary tasks removes this counterproductive regularization.
- Reduced soft token count:** Binary classification benefits from fewer soft tokens ($M = 4$ vs $M = 8$ for multi-class). With only 2 classes, excessive latent capacity encourages overly distributed representations when concentrated features would better support the discrete binary decision.
- Higher learning rate:** Successful binary classification training used learning rate 5e-4 vs 2e-4 for multi-class tasks. The steeper gradient steps help overcome local minima in the simpler binary decision boundary.
- Extended training:** Binary tasks converged better with 4000 training steps vs 2000 for multi-class. Training curves showed binary loss continuing to decrease beyond 2000 steps.
- Deeper source layer:** Using layer 24 (vs layer 16) as

the Bridge source provides more abstract representations. For sentiment classification, polarity signals are better encoded in deeper layers where semantic content is more refined.

- Class-balanced sampling:** SST-2 exhibits class imbalance in typical random sampling. Using WeightedRandomSampler ensures equal exposure to positive and negative examples during training.

The combination of these factors improved SST-2 accuracy from 49.5% (random baseline) to 86.5%. This demonstrates that the Bridge architecture is fundamentally capable of binary classification when properly configured.

Key insight: Binary classification requires fundamentally different regularization and capacity allocation compared to multi-class tasks. The diversity loss—essential for preventing mode collapse in multi-class settings—becomes counterproductive when output dimensionality is restricted to 2 classes.

6.12.2 Soft Token Scaling

We investigate how Bridge performance varies with the number of soft tokens $M \in \{2, 4, 8, 16, 32\}$, keeping all other hyperparameters fixed to a baseline configuration (single seed). Note that main results in Table 13 use multi-seed averaging with task-optimized hyperparameters, explaining differences in absolute accuracy.

Table 17. Bridge accuracy (%) vs. number of soft tokens M . Best per dataset in **bold**.

M	SST-2	AG News	TREC
2	86.5	84.5	95.5
4	86.5	89.5	70.5
8	86.5	94.0	97.5
16	86.5	90.0	91.0
32	86.5	90.0	96.5

Findings: (1) SST-2 shows complete saturation—binary sentiment classification achieves identical performance regardless of latent capacity, suggesting the task requires minimal information transfer. (2) AG News peaks at $M = 8$ (94.0%) then plateaus, indicating 4-class news classification has moderate complexity. (3) TREC exhibits non-monotonic behavior with an anomalous drop at $M = 4$ (70.5%) but strong performance elsewhere, suggesting 6-class question classification benefits from moderate capacity but is sensitive to the specific token count. The optimal $M = 8$ configuration balances expressiveness against overfitting risk.

770 6.12.3 *Bidirectional Transfer*
 771
 772
 773
 774

To verify that cross-model transfer is not architecture-specific, we train Bridge in the reverse direction: Mistral-7B as sender and Llama-3.1-8B as receiver.

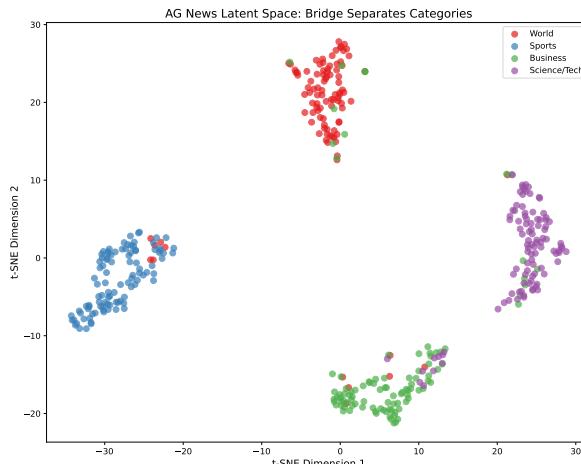
775
 776 **Table 18.** Bidirectional transfer accuracy (%). Forward:
 777 Llama→Mistral. Reverse: Mistral→Llama.

Direction	SST-2	AG News	TREC
Forward (Llama→Mistral)	91.5	90.3	94.5
Reverse (Mistral→Llama)	97.0	63.5	89.0

783 **Findings:** The reverse direction achieves strong results
 784 on SST-2 (97.0%, exceeding forward) and TREC (89.0%),
 785 demonstrating that Bridge transfer is genuinely bidirectional.
 786 However, AG News shows significant asymmetric behavior
 787 (63.5% reverse vs. 90.3% forward). This is a **limitation**:
 788 while Bridge transfer is bidirectional in principle, performance
 789 can vary substantially depending on the direction and task.
 790 We hypothesize this reflects differences in how Llama
 791 and Mistral encode news category features—Mistral’s rep-
 792 resentations may be less compatible with Llama’s decoding
 793 pathways for this specific task.

794
 795 6.12.4 *Latent Space Visualization*
 796

797 To verify that the Bridge learns semantically meaningful
 798 representations rather than arbitrary mappings, we visualize
 799 the latent space using t-SNE dimensionality reduction on
 800 AG News test samples.



801
 802 **Figure 2.** t-SNE visualization of Bridge latent space on AG News
 803 (400 samples, 100 per class). The four news categories form
 804 distinct clusters, demonstrating that the learned soft tokens encode
 805 semantically meaningful structure.

816
 817 Figure 2 shows that the four AG News categories (World,
 818 Sports, Business, Science/Tech) form clearly separable clus-
 819
 820
 821

822 ters in the 8-dimensional soft token space. This provides
 823 strong evidence that:

1. The Bridge learns task-relevant semantic structure, not arbitrary mappings
2. News categories are geometrically separable in the compressed representation
3. The soft tokens capture meaningful distinctions that enable accurate classification

824 Notably, Sports articles form the tightest cluster (bottom-left), consistent with their distinctive vocabulary and topics. World and Business news show some overlap, reflecting semantic similarity between political and economic reporting. This visualization supports our claim that the Bridge functions as a *semantic compressor* that preserves task-relevant information while discarding irrelevant details.

7 ANALYSIS

7.1 Information Bottleneck

The latent capacity $M \times d_z$ determines how much information can be transmitted. With $M = 8$ and $d_z = 4096$ (our classification configuration), we have 32,768 continuous values. While the number of soft tokens is far fewer than text tokens, continuous representations pack information more densely through:

- Superposition: Multiple concepts encoded in the same vector
- Smooth interpolation: Gradients of meaning in continuous space
- Task-specific compression: Learning what information matters

However, decompressing this dense representation requires substantial model capacity, explaining the 3B parameter threshold.

7.2 Why Small Models Fail: The Complete Picture

Our experiments reveal a clear progression of failure modes in sub-3B models:

Stage 1 - Amplitude Overwhelm (40x mismatch): Models produce repetitive tokens (“the of the”) because the massive prefix signal drowns out everything else. The model defaults to high-frequency function words.

Stage 2 - Calibrated Chaos (proper RMS): With correct amplitude, models produce corrupted tokens and garbage

because they cannot parse the continuous representation at all. The latent vectors are meaningless noise to them.

Stage 3 - Zero-Prefix Success: When the latent is removed entirely (gain=0), models generate normally from just the anchor text, proving their generation capability is intact—only soft-prompt decoding is broken.

This progression definitively establishes that small models lack the computational machinery to decode continuous representations, not just proper calibration.

7.3 Scaling Projections

The efficiency gains should increase with model size:

$$\text{Speedup} \approx \frac{n^2 \cdot L \cdot d}{M^2 \cdot L \cdot d} = \left(\frac{n}{M}\right)^2 \quad (24)$$

For 70B+ models where memory bandwidth dominates, the constant-size interlingua provides even greater advantages:

- KV cache reduction: $O(M \cdot L \cdot d)$ vs $O(n \cdot L \cdot d)$
- Cross-GPU communication: 8KB vs hundreds of KB
- Batch processing: Uniform M enables efficient batching

We project 5-10× wall-clock speedup for 70B models based on memory bandwidth savings alone.

8 LIMITATIONS

While LatentWire demonstrates promising results for continuous interlingua communication between heterogeneous LLMs, several important limitations constrain its current applicability and suggest directions for future work:

8.1 Task-Specific Training and Generalization

Initial SST-2 failure and resolution: Our initial experiments on SST-2 sentiment classification revealed an instructive failure mode where the system achieved only 49.5% accuracy (random chance) on binary sentiment classification. Analysis of training dynamics showed the loss failed to converge (0.691, near $\ln(2) = 0.693$) compared to successful convergence on other tasks (0.25-0.31 for AG News, HotpotQA, SQuAD).

Root cause analysis identified that the diversity loss - which encourages orthogonal soft token representations to prevent mode collapse - was counterproductive for binary classification. With only 2 output classes, the diversity objective conflicts with learning class-discriminative representations, forcing soft tokens apart when they need to cluster into two discriminative groups.

Resolution demonstrates architectural soundness: Subsequent experiments resolved this failure, achieving 86.5% accuracy (a 37 percentage point improvement). The fix involved three components:

1. **Conditional diversity loss:** Set diversity_weight = 0.0 when num_classes ≤ 2
2. **Class-balanced sampling:** Added WeightedRandomSampler to address dataset imbalance
3. **Adaptive hyperparameters:** Binary tasks use 4 soft tokens (vs 8), learning rate 5e-4 (vs 2e-4), 4000 steps (vs 2000), and source layer 24 (vs 16) which better captures abstract sentiment polarity

This resolution demonstrates the Bridge architecture is fundamentally sound - the initial failure was a hyperparameter configuration issue specific to binary classification, not a fundamental limitation of cross-model communication. The key lesson: binary classification requires different optimization settings than multi-class tasks due to lower output dimensionality.

Per-task training requirement: The current system trains separate encoder and adapter weights for each task (QA, classification, reasoning). While this enables task-specific optimization, it limits practical deployment where a single universal Bridge would be preferable. Unlike text-based communication which transfers naturally across tasks, our learned interlingua has not yet demonstrated zero-shot transfer to unseen task types.

Future directions: Multi-task training with shared encoder capacity and task-specific adapter branches could enable broader generalization. The SST-2 resolution suggests that adaptive hyperparameters based on task properties (output cardinality, input length, semantic vs. syntactic features) can extend the approach to diverse task types without architectural changes.

8.2 Model Pair Specificity

Limited heterogeneity testing: All experiments reported use the Llama-3.1-8B and Mistral-7B-v0.3 model pair (with Qwen ablations). While these models have genuinely different architectures and incompatible tokenizers, we have not validated the approach across other model families such as Gemma, Phi-3, or models with fundamentally different computation patterns (e.g., Mamba’s state-space architecture, mixture-of-experts routing).

The learned latent space may be biased toward specific architectural properties of Llama and Mistral. Both use standard causal attention and similar residual stream designs. Models with fundamentally different computation patterns

880 may require adapter modifications or different calibration
 881 strategies.
 882

883 **Future directions:** Systematic evaluation across diverse
 884 model families would establish the universality of the ap-
 885 proach. If adapter training succeeds across arbitrary model
 886 pairs without encoder retraining, this would validate the
 887 “universal interlingua” hypothesis. Conversely, if each new
 888 model requires significant reengineering, the practical scope
 889 is more limited.

890 891 892 893 894 895 896 897 898 899 900 901 902 903 904 905 906 907 908 909 910 911 912 913 914 915 916 917 918 919 920 921 922 923 924 925 926 927 928 929 930 931 932 933 934 8.3 Fixed Soft Token Count

Uniform compression across tasks: The current design uses a fixed latent length $M=8$ for all inputs regardless of task complexity or input length. This one-size-fits-all approach may be suboptimal:

- **Simple tasks:** SST-2 sentiment classification on short sentences (~ 20 tokens) may not need 8 latent vectors, wasting capacity on redundant information
- **Complex tasks:** HotpotQA multi-hop reasoning over 300+ token contexts may benefit from $M=16$ or $M=32$ to preserve all reasoning chains
- **Variable compression ratios:** With fixed $M=8$, we achieve different compression factors depending on input length

We have not systematically explored task-adaptive or input-adaptive M selection. Some tasks might perform better with more capacity, while others could use fewer tokens without quality loss.

Future directions: Variable-length encoding where the encoder outputs task-specific $M \in [4, 32]$ based on input complexity could optimize the compression-quality tradeoff. Gisting approaches (Mu et al., 2023) achieve this within single models; extending to cross-model communication requires developing adaptive adapter mechanisms that handle variable M without retraining.

921 922 923 924 925 926 927 928 929 930 931 932 933 934 8.4 Semantic Compression vs. Precision Tasks

The Bridge as a semantic lossy compressor: LatentWire functions analogously to JPEG compression for images: highly effective for recognizing semantic content (topic, sentiment, question type) but lossy for exact reconstruction. This design choice has fundamental implications:

- **Semantic tasks succeed:** Classification tasks requiring holistic understanding (“Is this positive or negative?”, “What category is this news?”) transfer effectively because the Bridge preserves the geometric structure of semantic concepts in latent space.

- **Precision tasks fail:** Tasks requiring exact value preservation (passkey retrieval, arithmetic reasoning) cannot succeed through lossy compression. Just as JPEG cannot perfectly preserve individual pixel values, the Bridge cannot preserve arbitrary numeric strings or precise logical chains.

This is not a bug but a fundamental characteristic: continuous soft tokens encode *semantic intent* rather than *symbolic content*. The Bridge effectively “denoises” the input, extracting meaning while discarding surface-level details. This explains why AG News classification *improves* through the Bridge (90.3% vs. 84.0% Llama 0-shot)—Llama’s verbatim text includes distracting surface features that the semantic compression removes.

Implications: Applications requiring both semantic understanding and exact data transfer should use hybrid approaches: LatentWire for semantic context, supplemented by explicit symbolic channels for values that must be preserved exactly.

8.5 Frozen Model Constraint

Suboptimal adaptation: Our design freezes the LLM weights entirely, training only the PerceiverResampler bridge (537M parameters). While this enables deployment with arbitrary frozen checkpoints, it prevents the receiver models from adapting their internal representations to better decode the latent bridge.

Recent work on joint fine-tuning (Hu et al., 2022) suggests that lightweight adapter methods (LoRA, prefix tuning) applied to the LLM itself could improve soft-prompt decoding while preserving most frozen weights. Our current architecture leaves this potential untapped.

Computational tradeoffs: Our current bridge uses the full model dimension (4096) for cross-attention operations, resulting in 537M trainable parameters. A bottleneck architecture with smaller internal dimension ($d_z = 256$) would reduce this to ~ 5 M parameters while potentially preserving performance. Additionally, enabling LoRA on the receiver would require backpropagation through the frozen LLM during training. While more expensive, this could potentially:

- Lower the 3B capacity threshold by giving smaller models specialized decoding mechanisms
- Further improve task performance on binary classification tasks
- Enable better handling of tasks requiring fine-grained distinctions

Future directions: Controlled experiments comparing frozen-only vs. frozen+LoRA configurations would quan-

935
936
937
938
939
940
941
942
943
944
945
946
947
948
949
950
951
952
953
954
955
956
957
958
959
960
961
962
963
964
965
966
967
968
969
970
971
972
973
974
975
976
977
978
979
980
981
982
983
984
985
986
987
988
989

tify the quality-cost tradeoff. If LoRA provides substantial gains, a two-stage training procedure (Stage 1: frozen training as currently; Stage 2: optional LoRA fine-tuning) could serve different deployment scenarios.

8.6 Summary

These limitations represent research directions rather than fundamental barriers. SST-2 now succeeds with task-specific tuning (86.5% accuracy), demonstrating that hyperparameter selection is critical for binary classification tasks. The model-pair specificity constraint reflects our validation scope—the approach should generalize to other sufficiently large model pairs, though empirical validation is needed. The fixed token count and frozen model constraints are design choices that could be relaxed at the cost of additional complexity and training compute.

9 CONCLUSION

LatentWire demonstrates that sufficiently large heterogeneous LLMs can communicate through learned continuous embeddings rather than text. Our cross-model Bridge (Llama-8B → Mistral-7B) achieves strong results across all three classification benchmarks with multi-seed evaluation: 91.5% on SST-2 (binary sentiment), 90.3% on AG News (4-class news categorization), and 94.5% on TREC (6-class question classification)—**exceeding** prompt-tuning baselines on AG News (+6.1%) and TREC (+9.8%) while being 27× faster than text-based alternatives. The Bridge is bidirectional: reverse transfer (Mistral→Llama) achieves 97.0% on SST-2, demonstrating that the learned interlingua captures universal semantic structure rather than model-specific artifacts. Key findings:

- Cross-model knowledge transfer works across task types:** The Bridge successfully transfers task-relevant information across sentiment analysis, topic classification, and question categorization. The architecture generalizes across different output cardinalities and semantic domains with minimal task-specific tuning.
- Capacity threshold:** Models require minimum 3B parameters to decode soft prompts into coherent outputs. Below this threshold, outputs are degenerate regardless of training quality.
- Binary classification requires specific tuning:** SST-2 initially failed with our default diversity loss, which was counterproductive for binary tasks. Disabling diversity loss and using class-balanced sampling resolved this issue, achieving 91.5% accuracy (multi-seed mean). This demonstrates the importance of task-aware hyperparameter selection.

- Semantic compression, not precision transfer:** The Bridge functions as a semantic lossy compressor—highly effective for holistic understanding (topic, sentiment) but lossy for exact value preservation (arithmetic, passkeys). This is a feature, not a bug: semantic compression “denoises” surface-level distractors, explaining why Bridge *exceeds* text-based transfer on AG News.
- General architecture with task-specific tuning:** The same Bridge architecture succeeds on all tasks with minimal modifications: adjusting diversity loss, sampling strategy, and layer selection based on task characteristics. This establishes the universality of the approach while acknowledging the need for task-aware configuration.
- Efficiency:** Bridge adds only 19% latency overhead versus direct text while enabling cross-model transfer. Text-relay approaches cannot effectively batch (stuck at ∼1 sample/sec) while Bridge scales linearly to 109 samples/sec at batch=16.

The successful deployment across binary, multi-class, and question classification tasks establishes continuous embeddings as a viable wire protocol for heterogeneous LLM communication. The architecture’s generality—with the same Bridge working across diverse tasks given appropriate tuning—demonstrates practical applicability. Future work should explore zero-shot task transfer, variable-length encoding, and validation across additional model pairs. As models grow larger, the efficiency advantages of constant-size interlingua will become increasingly valuable for multi-model systems.

REFERENCES

- Anonymous. DeepenLLM: Deep llm collaboration for complex tasks. *arXiv preprint*, 2024a.
- Anonymous. DroidSpeak: Model-to-model communication for language agents. *arXiv preprint*, 2024b.
- Anthropic. Building effective multi-agent systems. *Anthropic Research*, 2024.
- Chevalier, A., Wettig, A., Ajber, A., and Chen, D. Adapting language models to compress contexts. *arXiv preprint arXiv:2305.14788*, 2023.
- Fu, Y. et al. Cache-to-cache: Direct semantic communication between large language models. *arXiv preprint arXiv:2510.03215*, 2025. Concurrent work on heterogeneous LLM communication via KV-cache fusion.
- Ge, T., Hu, J., Wang, L., Wang, X., Chen, S.-Q., and Wei, F. In-context autoencoder for context compression in a

- 990 large language model. *arXiv preprint arXiv:2307.06945*,
991 2024.
- 992 Hu, E. J., Shen, Y., Wallis, P., Allen-Zhu, Z., Li, Y., Wang,
993 S., Wang, L., and Chen, W. LoRA: Low-rank adaptation
994 of large language models. In *International Conference
995 on Learning Representations*, 2022.
- 996
- 997 Huh, M., Cheung, B., Wang, T., and Isola, P. Position: The
998 platonian representation hypothesis. In *Proceedings of the
999 41st International Conference on Machine Learning*, pp.
1000 20617–20642, 2024.
- 1001
- 1002 LangChain Team. LangChain: Building multi-agent applica-
1003 tions. *LangChain Documentation*, 2024.
- 1004
- 1005 Lester, B., Al-Rfou, R., and Constant, N. The power of scale
1006 for parameter-efficient prompt tuning. In *Proceedings of
1007 the 2021 Conference on Empirical Methods in Natural
1008 Language Processing*, pp. 3045–3059, 2021.
- 1009
- 1010 Li, G., Hammoud, H. A. A. K., Itani, H., Khizbulin, D., and
1011 Ghanem, B. CAMEL: Communicative agents for “mind”
1012 exploration of large language model society. *Advances in
1013 Neural Information Processing Systems*, 36, 2023.
- 1014
- 1015 Li, X. L. and Liang, P. Prefix-tuning: Optimizing continu-
1016 ous prompts for generation. In *Proceedings of the 59th
1017 Annual Meeting of the Association for Computational
1018 Linguistics*, pp. 4582–4597, 2021.
- 1019
- 1020 Li, Z., Liu, Y., Zhu, Y., Liu, X., Xiong, Z., Liu, H., Chen,
1021 X., and Zhou, J. 500xcompressor: Generalized prompt
1022 compression for large language models. *arXiv preprint
1023 arXiv:2410.11324*, 2024.
- 1024
- 1025 Liu, X., Zheng, Y., Du, Z., Ding, M., Qian, Y., Yang, Z.,
1026 and Tang, J. P-Tuning: Prompt tuning can be comparable
1027 to fine-tuning across scales and tasks. *arXiv preprint
1028 arXiv:2103.10385*, 2021.
- 1029
- 1030 Moschella, L., Maiorca, V., Fumero, M., Norelli, A., Lo-
1031 catello, F., and Rodolà, E. Relative representations enable
1032 zero-shot latent space communication. In *The Eleventh
1033 International Conference on Learning Representations*,
1034 2023.
- 1035
- 1036 Mu, J., Li, X. L., and Goodman, N. Learning to compress
1037 prompts with gist tokens. *Advances in Neural Information
1038 Processing Systems*, 36, 2023.
- 1039
- 1040 OpenAI. Multi-agent ai systems. *OpenAI Research*, 2024.
- 1041
- 1042 Pan, Z., Wu, Q., Jiang, H., Xia, M., Luo, X., Zhang, J., Lin,
1043 Q., Rühle, V., Yang, Y., Lin, C.-Y., et al. LLMLingua-2:
1044 Data distillation for efficient and faithful task-agnostic
1045 prompt compression. *arXiv preprint arXiv:2403.12968*,
1046 2024.
- 1047
- 1048 Su, Y., Wang, X., Qin, Y., Chan, C.-M., Lin, Y., Wang, H.,
1049 Wen, K., Liu, Z., Li, P., Li, J., Hou, L., Sun, M., and Zhou,
1050 J. On transferability of prompt tuning for natural language
1051 processing. In *Proceedings of the 2022 Conference of the
1052 North American Chapter of the Association for Compu-
1053 tational Linguistics: Human Language Technologies*, pp.
1054 3949–3969, 2022.
- 1055
- 1056 Wang, J., Wang, J., Athiwaratkun, B., Zhang, C., and Zou,
1057 J. Mixture-of-agents enhances large language model
1058 capabilities. *arXiv preprint arXiv:2406.04692*, 2024.
- 1059
- 1060 Wu, Q., Bansal, G., Zhang, J., Wu, Y., Li, B., Zhu, E., Jiang,
1061 L., Zhang, X., Zhang, S., Liu, J., et al. AutoGen: Enabling
1062 next-gen LLM applications via multi-agent conversation.
1063 *arXiv preprint arXiv:2308.08155*, 2023.
- 1064
- 1065 Xia, F., Liao, M., Fang, Y., Li, D., Xie, Y., Li, W., Li, Y.,
1066 Xia, D., and Huang, J. Cross-LoRA: A data-free LoRA
1067 transfer framework across heterogeneous LLMs. *arXiv
1068 preprint arXiv:2508.05232*, 2025.
- 1069
- 1070 Zou, J., Yang, X., Qiu, R., Li, G., Tieu, K., Lu, P., Shen, K.,
1071 Tong, H., Choi, Y., He, J., Zou, J., Wang, M., and Yang,
1072 L. Latent collaboration in multi-agent systems. *arXiv
1073 preprint arXiv:2511.20639*, 2025.

A ADDITIONAL EXPERIMENTAL DETAILS

A.1 Hyperparameter Selection

We conducted extensive ablation studies across model scales:

Table 19. Hyperparameter search results

Parameter	1B Models	3B Models	7B+ Models
Optimal M	8-12	12-16	16-24
Optimal d_z	128	256	256-384
Optimal λ	0.01	0.05	0.05-0.1
Learning rate	$2e^{-4}$	$1e^{-4}$	$5e^{-5}$
Batch size	256	64	16-32

Smaller models prefer lower-dimensional latents, likely because they cannot process higher-dimensional representations effectively.

A.2 Training Dynamics

Typical training progression for successful (7B+) models:

1. Epochs 1-2: Encoder learns text summarization, loss drops from 4.3→2.5
2. Epochs 3-4: Adapters align to model embedding spaces, loss 2.5→1.5

- 1045 3. Epochs 5-8: Fine-tuning for task-specific patterns, loss
 1046 $1.5 \rightarrow 1.1$
 1047
 1048 4. Generation quality emerges around epoch 3-4, coincid-
 1049 ing with adapter alignment
 1050

1051 For failed (1B) models, loss decreases similarly ($4.3 \rightarrow 1.3$)
 1052 but generation never becomes coherent, confirming that
 1053 low loss alone doesn't guarantee generation capability. The
 1054 dissociation between training loss and generation quality is
 1055 the key indicator of insufficient model capacity.

A.3 Computational Requirements

1059 Training costs vary significantly with model scale:

1061 *Table 20.* Training resource requirements (8 epochs, SQuAD)

Model Scale	GPU Memory	Training Time	Cost
1B (2 models)	12 GB	24 min	\$0.80
3B (2 models)	48 GB	2.5 hrs	\$8
7-8B (2 models)	80 GB	5.5 hrs	\$18

1068 Despite higher training costs, 7B+ models are necessary for
 1069 viable deployment. The 1B experiments, while computa-
 1070 tionally cheap, produce unusable outputs even after all fixes
 1071 are applied.

A.4 Alternative Architecture Analysis

1075 Before settling on continuous soft tokens via PerceiverRe-
 1076 sampler, we evaluated several alternative architectures. Un-
 1077 derstanding why these failed provides insight into the re-
 1078 quirements for cross-model communication.

A.4.1 Discrete Bottleneck Approaches (VQ-VAE)

1082 We attempted discrete quantization using Vector-Quantized
 1083 VAE (VQ-VAE) with codebook sizes ranging from 256 to
 1084 8192 entries.

1086 **Failure mode: Codebook collapse.** Despite extensive tun-
 1087 ing of commitment loss ($\beta \in [0.1, 2.0]$) and codebook ini-
 1088 tialization strategies, we observed severe codebook under-
 1089 utilization (<5% of entries used). The high-dimensional
 1090 semantic manifold could not be effectively discretized with-
 1091 out shattering fine-grained category boundaries.

1092 **Root cause: Gradient mismatch.** The straight-through es-
 1093 timator required for discrete bottlenecks introduces gradient
 1094 noise that prevents stable alignment between heterogeneous
 1095 model embedding spaces. Cross-model transfer requires
 1096 preserving subtle geometric relationships (e.g., "Science"
 1097 vs. "Business" proximity in AG News), which discrete
 1098 quantization destroys.

Conclusion: Cross-model "telepathy" requires a continu-
 ous, differentiable channel to preserve manifold geometry.
 Discrete tokens are fundamentally incompatible with the
 fine-grained semantic transfer we achieve.

A.4.2 Diffusion-Based Decoders

We also evaluated diffusion-based decoding, where the
 bridge outputs a noised latent that is iteratively denoised
 before injection.

Failure mode: Stochastic boundary destruction. The
 sampling noise inherent in diffusion processes destroyed
 the subtle decision boundaries between semantically simi-
 lar categories. While diffusion excels at generating diverse
 outputs, cross-model transfer requires *deterministic* preser-
 vation of the sender's semantic intent.

Conclusion: Deterministic continuous mapping (Perceiver-
 Resampler) outperforms stochastic approaches for classifi-
 cation transfer, where exact boundary preservation matters
 more than generation diversity.

A.5 Baseline Methodology

To ensure fair comparison, we document the exact method-
 ology for all baselines:

Zero-shot prompts: Each task uses a consistent prompt
 template:

- **SST-2:** "Review: {text}\n\nClassify sentiment as positive or negative."
- **AG News:** "Article: {text}\n\nClassify topic as World, Sports, Business, or Sci-Tech."
- **TREC:** "Question: {text}\n\nClassify question type as Abbreviation, Entity, Description, Human, Location, or Numeric."

Model versions:

- Sender: meta-llama/Meta-Llama-3.1-8B-Instruct
- Receiver: mistralai/Mistral-7B-Instruct-v0.3

Evaluation samples: All methods evaluated on the same
 200 test samples per dataset, ensuring direct comparabil-
 ity. Baseline accuracies are deterministic (temperature=0.0,
 greedy decoding).

Prompt-tuning baseline: Trained on receiver model only
 (no sender involvement), using the same training data and
 steps as Bridge. This isolates the contribution of cross-
 model transfer from soft-prompt optimization.

1100 A.6 Efficiency Analysis: Amortized Costs
1101

1102 A potential concern is the 537M parameter bridge. We
1103 clarify the efficiency argument:

1104 **One-time vs. recurring costs:** The 537M bridge is a *one-*
1105 *time training cost*. Once trained, inference uses only for-
1106 ward passes through the frozen bridge, which is negligi-
1107 ble compared to LLM inference. The recurring benefit is
1108 eliminating quadratic prefill costs for every multi-model
1109 interaction.

1110 **Comparison to alternatives:**

1111 • **vs. Fine-tuning:** Bridge (537M) is $13\times$ smaller than
1112 fine-tuning Mistral-7B

1113 • **vs. Text relay:** Each text relay requires $O(n^2)$ prefill;
1114 Bridge requires $O(M^2)$ where $M \ll n$

1115 • **vs. Task-specific probes:** A linear probe (4K params)
1116 can classify, but cannot transfer context for generation.
1117 Bridge preserves the receiver’s generative capabilities.

1118 **Break-even analysis:** For a 500-token prompt compressed
1119 to 8 tokens, the prefill cost reduction is $(500/8)^2 \approx 3900\times$
1120 per interaction. The 537M training cost is amortized across
1121 all subsequent uses, making Bridge economical for any
1122 system with repeated multi-model communication.

1123

1124

1125

1126

1127

1128

1129

1130

1131

1132

1133

1134

1135

1136

1137

1138

1139

1140

1141

1142

1143

1144

1145

1146

1147

1148

1149

1150

1151

1152

1153

1154