# TELEPATHY: CROSS-MODEL COMMUNICATION VIA SOFT TOKENS

**Anonymous Authors**[1]


## ABSTRACT

We present Telepathy, a method for enabling communication between heterogeneous large language models (LLMs) through learned soft tokens, bypassing autoregressive text generation entirely. Our approach uses a lightweight Perceiver Resampler bridge (188K parameters) to transform hidden states from a sender model (Llama 3.1 8B) into soft tokens that directly condition a receiver model (Mistral 7B). On text classification benchmarks, Telepathy achieves **22.4× lower latency** than text-relay baselines (37ms vs. 835ms) while maintaining or exceeding task accuracy. Critically, we show that the sender model is essential: prompt-tuning on Mistral alone achieves only random chance (49.5% on SST-2), while the bridge achieves **96.7%**—a 47pp improvement from Llama's hidden states. We observe **super-additive performance**: the bridge exceeds both Llama (92.0%) and Mistral (88.5%) operating independently. Our experiments reveal an inverse scaling law where fewer soft tokens (8-16) consistently outperform more tokens (32-128). These findings demonstrate that cross-model communication via continuous representations can be both faster and more effective than discrete text.


## 1 INTRODUCTION

Large language models (LLMs) have emerged as powerful tools for natural language understanding and generation (Vaswani et al., 2017; Touvron et al., 2023; Jiang et al., 2023a). However, the dominant paradigm for combining multiple LLMs involves sequential text generation: one model produces text that another model consumes. This approach incurs substantial latency due to autoregressive decoding and may lose information through the discretization bottleneck of natural language.

We propose **Telepathy**, a method that enables direct communication between heterogeneous LLMs through learned soft tokens. Rather than having a sender model generate text for a receiver model to process, Telepathy transforms the sender's internal representations into a small set of continuous embeddings (soft tokens) that directly condition the receiver model's inference. This approach:

1. **Eliminates autoregressive generation latency**: The sender model only performs a single forward pass, reducing end-to-end latency by over 20× compared to text-relay approaches.
2. **Preserves continuous information**: Soft tokens can encode nuances that may be lost when discretizing to natural language tokens.
3. **Enables super-additive performance**: The combined system can outperform either model operating independently, suggesting emergent capabilities from cross-model communication.

Our key contributions are:

- A lightweight bridge architecture based on Perceiver Resampler (Jaegle et al., 2021; Alayrac et al., 2022) that transforms hidden states between heterogeneous LLMs with only 188K trainable parameters.
- Comprehensive evaluation across four text classification benchmarks (SST-2, AG News, TREC, Banking77) demonstrating consistent improvements over text baselines.
- Analysis of an inverse scaling phenomenon where compression to fewer soft tokens improves rather than degrades performance.
- Latency benchmarks showing 22.4× speedup over text-relay and 2.6× speedup over direct text inference.

## 2 RELATED WORK

**Soft Prompts and Prompt Tuning** Prompt tuning (Lester et al., 2021) and prefix tuning (Li & Liang, 2021) demonstrated that freezing LLM weights while learning continuous "soft" prompt embeddings can match full fine-tuning performance. Our work extends this paradigm from single-model adaptation to cross-model communication, using soft tokens as an interlingua between heterogeneous models.

[1]**AUTHORERR: Missing \mlsysaffiliation. . AUTHORERR: Missing \mlsyscorrespondingauthor.**

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

**Perceiver Architecture** The Perceiver (Jaegle et al., 2021) introduced cross-attention to map arbitrary-length inputs to a fixed-size latent array, enabling efficient processing of diverse modalities. Perceiver IO extended this to arbitrary outputs. Our bridge architecture draws from this design, using cross-attention to compress sender hidden states into a small number of soft tokens.

**Vision-Language Models** BLIP-2 (Li et al., 2023) introduced the Q-Former, a lightweight transformer that bridges frozen image encoders and frozen LLMs through learned query tokens. Flamingo (Alayrac et al., 2022) similarly used a Perceiver Resampler to map visual features to soft prompts for LLM conditioning. Our work applies similar architectural principles to bridge two language models rather than vision and language modalities.

**Model Stitching and Knowledge Transfer** Model stitching (Bansal et al., 2021; Pan et al., 2023) connects layers from different networks using learned transformations. Cross-LoRA (Anonymous, 2024) enables transferring LoRA adapters between heterogeneous models. Knowledge distillation (Hinton et al., 2015; Gu et al., 2024) transfers capabilities from large to small models. Our approach differs by enabling runtime communication between models rather than offline knowledge transfer.

**Multi-Agent LLM Systems** Recent work on multi-agent systems (Anonymous, 2025; Wu et al., 2023) explores collaboration between multiple LLMs through natural language communication. While effective, text-based communication incurs latency from autoregressive generation. Telepathy provides a faster alternative through continuous representations.

**Prompt Compression** Methods like LLMLingua (Jiang et al., 2023b) compress prompts by removing tokens while preserving task performance. Soft prompt methods like ICAE (Ge et al., 2024) and 500xCompressor (Li et al., 2024) learn to compress context into dense embeddings. Our work focuses on cross-model transfer rather than single-model compression.

# 3 Method

## 3.1 Problem Setup

Given a sender model $\mathcal{S}$ (Llama 3.1 8B) and a receiver model $\mathcal{R}$ (Mistral 7B), we aim to transmit task-relevant information from $\mathcal{S}$ to $\mathcal{R}$ without generating text. Both models remain frozen; only the bridge is trained.

Let $\mathbf{h}_\mathcal{S} \in \mathbb{R}^{L \times d_\mathcal{S}}$ denote the hidden states from layer $\ell$ of the sender, where $L$ is the sequence length and $d_\mathcal{S} = 4096$ is Llama's hidden dimension. We seek a function $f_\theta$ that maps $\mathbf{h}_\mathcal{S}$ to soft tokens $\mathbf{z} \in \mathbb{R}^{M \times d_\mathcal{R}}$ that can condition $\mathcal{R}$, where $M \ll L$ is the number of soft tokens and $d_\mathcal{R} = 4096$ is Mistral's embedding dimension.

## 3.2 Bridge Architecture

Our bridge uses a Perceiver Resampler design:

1. **Input Projection**: Linear projection from sender hidden dimension to bridge internal dimension: $\mathbf{h}' = \mathbf{W}_{\text{in}}\mathbf{h}_\mathcal{S}$, where $\mathbf{W}_{\text{in}} \in \mathbb{R}^{d_\mathcal{S} \times d}$.

2. **Learned Latent Queries**: A set of $M$ learnable query vectors $\mathbf{Q} \in \mathbb{R}^{M \times d}$ that attend to the projected sender states.

3. **Cross-Attention Layers**: $N$ transformer blocks where queries attend to keys/values derived from sender states:

$$\mathbf{z}^{(n+1)} = \text{FFN}(\text{CrossAttn}(\mathbf{z}^{(n)}, \mathbf{h}')) \tag{1}$$

   We use $N = 2$ layers with $d = 512$ internal dimension.

4. **Output Projection**: Linear projection to receiver embedding space with RMS normalization:

$$\mathbf{z} = \alpha \cdot \frac{\mathbf{W}_{\text{out}}\mathbf{z}^{(N)}}{\text{RMS}(\mathbf{W}_{\text{out}}\mathbf{z}^{(N)})} \tag{2}$$

   where $\alpha$ is calibrated to match the receiver's embedding statistics.

The total parameter count is approximately 188K, negligible compared to the frozen 8B+7B models.

## 3.3 Training Objective

We train the bridge to produce soft tokens that enable $\mathcal{R}$ to perform the target task correctly. For classification tasks, we use cross-entropy loss on the receiver's predictions:

$$\mathcal{L} = -\sum_c y_c \log p_\mathcal{R}(c|\mathbf{z}, \mathbf{x}_{\text{prompt}}) \tag{3}$$

where $y_c$ is the ground-truth label and $p_\mathcal{R}$ is the receiver's predicted probability given soft tokens $\mathbf{z}$ and a task prompt $\mathbf{x}_{\text{prompt}}$.

We also add a diversity regularization term to prevent mode collapse:

$$\mathcal{L}_{\text{div}} = -\lambda \cdot H(\bar{\mathbf{z}}) \tag{4}$$

where $H$ is entropy and $\bar{\mathbf{z}}$ is the mean soft token representation across the batch.

### 3.4 Inference Pipeline

At inference time:

1. **Sender Encode** (16.9ms): Pass input through frozen $\mathcal{S}$, extract layer $\ell$ hidden states.
2. **Bridge Transform** (1.2ms): Apply $f_\theta$ to obtain $M$ soft tokens.
3. **Receiver Decode** (19.3ms): Prepend soft tokens to task prompt, run single forward pass through $\mathcal{R}$.

Total latency: 37.3ms, compared to 834.5ms for text-relay.

## 4 Experiments

### 4.1 Setup

**Models** We use Llama 3.1 8B Instruct as the sender and Mistral 7B Instruct v0.3 as the receiver. Both models remain frozen throughout training.

**Datasets** We evaluate on four text classification benchmarks:

- **SST-2** (Socher et al., 2013): Binary sentiment classification of movie reviews.
- **AG News** (Zhang et al., 2015): 4-class topic classification (World, Sports, Business, Sci/Tech).
- **TREC** (Li & Roth, 2002): 6-class question type classification.
- **Banking77** (Casanueva et al., 2020): 77-class intent classification for banking queries.

**Baselines** We compare against:

- **Llama Direct**: Llama classifies directly from text (sender ceiling).
- **Mistral Direct**: Mistral classifies directly from text (receiver baseline).
- **Text-Relay**: Llama generates a summary, Mistral classifies from summary.
- **Prompt-Tuning**: Learnable soft prompts on Mistral only (no Llama). This critical baseline tests whether the sender model actually contributes.

**Hyperparameters** Default settings: $M = 8$ soft tokens, $\ell = 31$ (last layer), learning rate $10^{-4}$, batch size 8, diversity weight $\lambda = 0.1$, 2000 training steps. For Banking77 and TREC, we use $M = 16$ tokens and 3000 steps.

*Table 1.* Classification accuracy (%) across benchmarks. Bridge consistently outperforms all baselines. Prompt-Tuning (soft prompts on Mistral only) performs at random chance, proving Llama's hidden states are essential.

| Method | SST-2 | AG News | TREC | Bank77 |
|---|---|---|---|---|
| Random Chance | 50.0 | 25.0 | 16.7 | 1.3 |
| Prompt-Tuning | 49.5$\pm$0.0 | 19.8$\pm$9.2 | 19.0$\pm$6.1 | – |
| Llama Direct | 92.0 | 79.0 | 53.5 | 22.0 |
| Mistral Direct | 88.5 | 79.0 | 43.0 | 19.5 |
| Text-Relay | 71.0 | 64.5 | 58.0 | 1.0 |
| **Bridge (ours)** | **96.7**$\pm$0.8 | **88.9** | **94.5** | **21.5** |

### 4.2 Main Results

Table 1 presents our main accuracy comparison.

**Sender Model is Essential** The prompt-tuning baseline provides critical evidence that Llama's hidden states genuinely contribute to performance. When we train learnable soft prompts on Mistral alone (same training budget, no Llama involvement), accuracy equals random chance: 49.5% on SST-2 (vs. 50% random), 19.8% on AG News (vs. 25% random), and 19.0% on TREC (vs. 16.7% random). In contrast, the bridge achieves 96.7% on SST-2—a **+47.2pp improvement** solely from incorporating Llama's representations. This definitively shows that cross-model communication via hidden states, not merely training soft prompts, drives the performance gains.

**Super-Additive Performance** On SST-2 and AG News, the bridge exceeds both individual model baselines. On SST-2, the bridge achieves 96.7% vs. Llama's 92.0% (+4.7pp) and Mistral's 88.5% (+8.2pp). On AG News, the bridge reaches 88.9% vs. both models' 79.0% (+9.9pp). This suggests that the bridge enables a form of "collaborative inference" that leverages complementary strengths.

**Bridge vs. Text-Relay** The bridge outperforms text-relay by large margins: +23.7pp on SST-2, +24.4pp on AG News, +36.5pp on TREC, and +20.5pp on Banking77. Text-relay catastrophically fails on Banking77 (1.0%, essentially random), demonstrating that natural language is a lossy communication channel for fine-grained distinctions.

**TREC Results** On TREC, the bridge achieves 94.5%, dramatically exceeding Llama (53.5%) and Mistral (43.0%) by 41pp and 51.5pp respectively. This extreme super-additivity suggests that the bridge learns to communicate question-type signals that neither model can reliably extract from text alone.

*Table 2.* Latency comparison (ms) on H100 GPU. Bridge achieves 22.4× speedup over text-relay by avoiding autoregressive generation.

| Method | Latency (ms) | Speedup |
|---|---|---|
| Text-Relay | 834.5 | 1.0× |
| Mistral Direct | 98.8 | 8.4× |
| **Bridge (ours)** | **37.3** | **22.4×** |

*Table 3.* Effect of soft token count on Banking77 accuracy. Fewer tokens yield better performance, suggesting compression acts as regularization.

| Soft Tokens | Accuracy (%) |
|---|---|
| 16 | **21.5** |
| 32 | 13.5 |
| 64 | 7.5 |
| 128 | 1.0 |

### 4.3 Latency Analysis

Table 2 presents latency measurements on an NVIDIA H100 GPU.

The bridge is 22.4× faster than text-relay and 2.6× faster than direct Mistral inference. The speedup comes from eliminating autoregressive generation in the sender (Llama generate: 745ms vs. Llama encode: 17ms). Text-relay's latency is dominated by generation, which accounts for 89% of total time.

Bridge latency breakdown:

- Llama encode: 16.9ms (45%)
- Bridge transform: 1.2ms (3%)
- Mistral decode: 19.3ms (52%)

### 4.4 Inverse Token Scaling

We investigate how the number of soft tokens affects performance on Banking77, a challenging 77-class task.

Table 3 shows a striking inverse relationship: increasing tokens from 16 to 128 causes accuracy to collapse from 21.5% to random (1.3% for 77 classes). This "inverse scaling" phenomenon suggests:

1. **Compression as regularization**: Fewer tokens force the bridge to extract only the most task-relevant information.
2. **Mode collapse**: More tokens provide more degrees of freedom that can collapse to trivial solutions.
3. **Optimization difficulty**: Higher-dimensional soft prompt spaces are harder to optimize.

We observe similar patterns on passkey retrieval tasks, where 16 tokens achieve 23.4% digit accuracy vs. 9.8% for 128 tokens.

## 5 Analysis

### 5.1 Why Super-Additive Performance?

The super-additive results on SST-2, AG News, and TREC are surprising. We hypothesize several explanations:

**Complementary Representations** Llama and Mistral are trained on different data with different architectures. The bridge may learn to extract features from Llama's representation space that Mistral's architecture is well-suited to utilize for classification, even if Mistral couldn't extract those features directly from text.

**Denoising Effect** The bridge acts as an information bottleneck that filters out noise and irrelevant details, passing only task-relevant signals to the receiver.

**Implicit Ensemble** The system effectively creates an ensemble where Llama's understanding informs Mistral's decision, combining their capabilities without the information loss of text discretization.

### 5.2 Text-Relay Failure Modes

Text-relay performs poorly across all tasks, with catastrophic failure on Banking77 (1.0%). Analysis reveals:

1. **Information loss**: Summarization discards fine-grained details needed for 77-way classification.
2. **Vocabulary mismatch**: Llama's summaries may use phrasings that don't trigger correct classifications in Mistral.
3. **Error propagation**: Mistakes in summarization compound with mistakes in classification.

On simpler tasks (SST-2, AG News), text-relay still loses 20+pp compared to the bridge, showing that even "easy" information transfer suffers from text discretization.

### 5.3 Comparison with Prompt Compression

Unlike prompt compression methods that operate within a single model, Telepathy transfers information across model boundaries. This enables:

- **Heterogeneous model collaboration**: Different architectures (Llama, Mistral) can communicate.
- **Capability composition**: Combine a model good at understanding with one good at generation.
- **Parallel inference**: With appropriate scheduling, sender and receiver compute can overlap.

## 6 Limitations and Future Work

**Task-Specific Training** Currently, bridges must be trained per-task. Future work could explore universal bridges that transfer across tasks, potentially through meta-learning or larger bridge architectures.

**Generative Tasks** We focus on classification; extending to generation tasks (translation, summarization) requires additional investigation of how to condition decoder generation on soft tokens.

**More Model Pairs** We demonstrate Llama→Mistral transfer; future work should validate across more model families and sizes.

**Theoretical Understanding** Why does compression help? Why is performance super-additive? Deeper theoretical analysis could inform better architecture design.

## 7 Conclusion

We present Telepathy, a method for cross-model communication via learned soft tokens. Our lightweight bridge (188K parameters) enables a sender LLM to condition a receiver LLM's inference without text generation, achieving:

- **22.4× lower latency** than text-relay (37ms vs. 835ms)
- **Sender model is essential**: Prompt-tuning alone achieves random chance (49.5%), while Bridge achieves 96.7% (+47pp from Llama's hidden states)
- **Super-additive performance** on SST-2 (96.7% vs. 92%/88.5%) and AG News (88.9% vs. 79%/79%)
- **Inverse token scaling** where fewer soft tokens yield better performance

These results demonstrate that continuous representations can be a more efficient and effective communication channel between LLMs than discrete text. The prompt-tuning baseline definitively shows that the sender model's hidden states—not merely training—drive the performance gains. Telepathy opens new possibilities for building collaborative multi-model systems with lower latency and higher accuracy.

## References


Alayrac, J.-B., Donahue, J., Luc, P., Miech, A., Barr, I., Hasson, Y., Lenc, K., Mensch, A., Millican, K., Reynolds, M., et al. Flamingo: a visual language model for few-shot learning. In *Advances in Neural Information Processing Systems*, volume 35, pp. 23716–23736, 2022.

Anonymous. Cross-LoRA: A data-free LoRA transfer framework across heterogeneous LLMs. *arXiv preprint arXiv:2508.05232*, 2024.

Anonymous. Multi-agent collaboration mechanisms: A survey of LLMs. *arXiv preprint arXiv:2501.06322*, 2025.

Bansal, Y., Nakkiran, P., and Barak, B. Revisiting model stitching to compare neural representations. *Advances in Neural Information Processing Systems*, 34:225–236, 2021.

Casanueva, I., Temčinas, T., Gerz, D., Henderson, M., and Vulić, I. Efficient intent detection with dual sentence encoders. In *Proceedings of the 2nd Workshop on Natural Language Processing for Conversational AI*, pp. 38–45, 2020.

Ge, T., Hu, J., Wang, L., Wang, X., Chen, S.-Q., and Wei, F. In-context autoencoder for context compression in a large language model. *arXiv preprint arXiv:2307.06945*, 2024.

Gu, Y., Dong, L., Wei, F., and Huang, M. MiniLLM: Knowledge distillation of large language models. *arXiv preprint arXiv:2306.08543*, 2024.

Hinton, G., Vinyals, O., and Dean, J. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2015.

Jaegle, A., Gimeno, F., Brock, A., Zisserman, A., Vinyals, O., and Carreira, J. Perceiver: General perception with iterative attention. In *International Conference on Machine Learning*, pp. 4651–4664, 2021.

Jiang, A. Q., Sablayrolles, A., Mensch, A., Bamford, C., Chaplot, D. S., Casas, D. d. l., Bressand, F., Lengyel, G., Lample, G., Saulnier, L., et al. Mistral 7B. *arXiv preprint arXiv:2310.06825*, 2023a.

Jiang, H., Wu, Q., Lin, C.-Y., Yang, Y., and Qiu, L. LLMLingua: Compressing prompts for accelerated inference of large language models. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pp. 13358–13376, 2023b.

Lester, B., Al-Rfou, R., and Constant, N. The power of scale for parameter-efficient prompt tuning. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pp. 3045–3059, 2021.

Li, J., Li, D., Savarese, S., and Hoi, S. BLIP-2: Bootstrapping language-image pre-training with frozen image encoders and large language models. In *International Conference on Machine Learning*, pp. 19730–19742, 2023.

Li, X. and Roth, D. Learning question classifiers. In *COLING 2002: The 19th International Conference on Computational Linguistics*, 2002.

Li, X. L. and Liang, P. Prefix-tuning: Optimizing continuous prompts for generation. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics*, pp. 4582–4597, 2021.

Li, Z., Liu, Y., Zhu, Y., Liu, X., Xiong, Z., Liu, H., Chen, X., and Zhou, J. 500xcompressor: Generalized prompt compression for large language models. *arXiv preprint arXiv:2410.11324*, 2024.

Pan, Z., Cai, J., and Zhuang, B. Stitchable neural networks. *arXiv preprint arXiv:2302.06586*, 2023.

Socher, R., Perelygin, A., Wu, J., Chuang, J., Manning, C. D., Ng, A., and Potts, C. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pp. 1631–1642, 2013.

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

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., and Polosukhin, I. Attention is all you need. In *Advances in Neural Information Processing Systems*, volume 30, 2017.

Wu, Q., Bansal, G., Zhang, J., Wu, Y., Li, B., Zhu, E., Jiang, L., Zhang, X., Zhang, S., Liu, J., et al. AutoGen: Enabling next-gen LLM applications via multi-agent conversation. *arXiv preprint arXiv:2308.08155*, 2023.

Zhang, X., Zhao, J., and LeCun, Y. Character-level convolutional networks for text classification. In *Advances in Neural Information Processing Systems*, volume 28, 2015.

# A Additional Experimental Details

## A.1 Hardware and Training Time

All experiments were conducted on NVIDIA H100 80GB GPUs. Training times:

- SST-2/AG News (2000 steps): 3.5 minutes
- TREC (2000 steps): 3.5 minutes
- Banking77 (3000 steps): 5.0 minutes

Total training time for all bridge variants: approximately 42 minutes.

## A.2 Confidence Intervals

We use Wilson score intervals at 95% confidence. With 200 evaluation samples:

- SST-2 Bridge: 94.7% [90.9, 97.1]
- AG News Bridge: 88.9% [83.8, 92.5]
- TREC Bridge: 94.5% [90.6, 97.0]
- Banking77 Bridge: 21.5% [16.3, 27.7]

## A.3 Hyperparameter Sensitivity

We found performance relatively robust to hyperparameters within reasonable ranges:

- Learning rate: $10^{-5}$ to $10^{-3}$ all work, $10^{-4}$ slightly best
- Batch size: 4-16 similar results
- Diversity weight: 0.05-0.2 prevents mode collapse
- Source layer: Last layer (31) consistently best