# IPR: Intelligent Prompt Routing with User-Controlled Quality-Cost Trade-offs

**Aosong Feng**[*], **Balasubramaniam Srinivasan**[*], **Yun Zhou**[*], **Zhichao Xu**[*], **Kang Zhou** , **Sheng Guan**, **Yueyan Chen**, **Xian Wu**, **Ninad Kulkarni**, **Yi Zhang**, **Zhengyuan Shen**, **Dmitriy Bespalov**, **Soumya Smruti Mishra**, **Yifei Teng**, **Darren Yow-Bang Wang**, **Haibo Ding**[†], **Lin Lee Cheong**
AWS AI
{asfeng, srbalasu, yunzzhou, xzhichao, zhoukang, shguan, yyanc, xianwwu, ninadkul, imyi, donshen, dbespal, soumish, yifeit, ybwang, hbding, lcheong}@amazon.com

## Abstract

Routing incoming queries to the most cost-effective LLM while maintaining response quality poses a fundamental challenge in optimizing performance-cost trade-offs for large-scale commercial systems. We present IPR — a quality-constrained **I**ntelligent **P**rompt **R**outing framework that dynamically selects optimal models based on predicted response quality and user-specified tolerance levels. IPR introduces three key innovations: (1) a modular architecture with lightweight quality estimators trained on 1.5M prompts annotated with calibrated quality scores, enabling fine-grained quality prediction across model families; (2) a user-controlled routing mechanism with tolerance parameter $\tau \in [0, 1]$ that provides explicit control over quality-cost trade-offs; and (3) an extensible design using frozen encoders with model-specific adapters, reducing new model integration effort. To rigorously train and evaluate IPR, we detail our efforts to curate an IPR dataset containing 1.5 million examples with response quality annotations across 11 LLM candidates. Deployed on a major cloud platform, IPR achieves 43.9% cost reduction while maintaining quality parity with the strongest model in the Claude family and processes requests with consistently low latency. The deployed system and additional product details are publicly available at https://aws.amazon.com/bedrock/intelligent-prompt-routing/.

## 1 Introduction

The proliferation of large language models (LLMs) with varying capabilities and costs has created a fundamental challenge in production deployments: how to automatically route incoming queries to the most cost-effective model while maintaining acceptable response quality (Hu et al., 2024). This challenge is exemplified in multi-model platforms like Amazon Bedrock, where models range from lightweight options like Claude Haiku to state-of-the-art models like Claude-3.5-Sonnet — a 12x cost difference. User queries exhibit enormous diversity: simple factual questions can be handled by smaller models, while complex reasoning tasks require more performant ones (Jaech et al., 2024; Guo et al., 2025). However, existing systems either force users to manually select models, or employ rigid routing rules that fail to capture the continuous spectrum of query complexity, resulting in substantial unnecessary costs or degraded user experiences at scale (Lu et al., 2024; Ding et al., 2024; Ong et al., 2024).

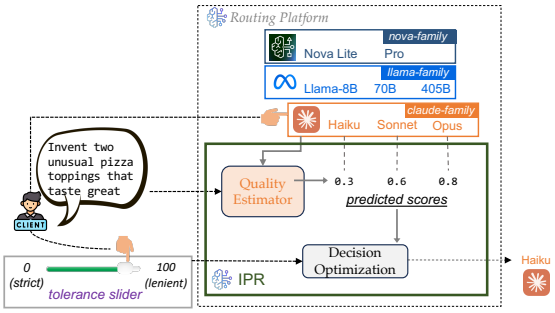


Figure 1: Routing prompt with IPR with user tolerance.

Deploying prompt routing onto real-world production systems poses several fundamental challenges which existing approaches have not addressed comprehensively. (1) **Quality prediction without response generation** requires routers to estimate response quality for each candidate model using only the input prompt, without actually generating responses — a challenging task given that model capabilities vary across different query types and domains. While recent works like RouteLLM (Ong et al., 2024) explored lightweight BERT-based classifiers for this prediction, they either support only binary strong/weak decisions rather than continuous quality estimation or are

[*]Equal contribution
[†]Corresponding author: hbding@amazon.com



*Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing: Industry Track*, pages 2484–2498
November 4-9, 2025 

trained on small-scale datasets. (2) **Latency constraints** in production systems prevent use of approaches that require multiple model invocations or complex computations; cascade-based methods circumvent this by sequential evaluation but sacrifice flexibility in model selection (Yue et al., 2024; Chen et al., 2023). (3) **Model extensibility and diversity** becomes critical as production platforms must simultaneously support diverse model families (e.g., Nova, Claude, Llama) each with distinct characteristics, while seamlessly integrating frequent model updates and releases. Most existing routers require complete retraining when model portfolios change (Lu et al., 2024; Ding et al., 2024), making them impractical for dynamic environments like a centralized LLM inference platform where new model versions appear monthly. (4) **User-specific quality-cost preferences** vary significantly across applications — a financial analysis task may prioritize accuracy while a chatbot may favor cost efficiency — yet current routing solutions offer little user control, typically hard-coding fixed quality thresholds (Ding et al., 2024) that cannot adapt to diverse business requirements.

To tackle these challenges, we propose **Intelligent Prompt Routing (IPR)**, a quality-constrained framework that dynamically selects the most cost-effective model while satisfying user-specified quality requirements. Our contributions are:

- **Industrial-scale quality prediction**: IPR trains neural estimators on 1.5M prompts annotated with calibrated reward scores from all candidate models, enabling fine-grained quality estimation that achieves 43.9% cost reduction in Claude family while maintaining quality parity.
- **Efficient extensible architecture**: A modular design with backbone prompt encoder and lightweight candidate-specific adapters enables fast routing decisions.
- **User-controlled routing**: A quality tolerance parameter ($\tau \in [0, 1]$) provides explicit control over cost-quality trade-offs, from maximum quality ($\tau = 0$) to aggressive savings ($\tau = 1$), with dynamic per-prompt threshold adjustment.

## 2 Problem Formulation and Routing Framework

We present a formal treatment of the prompt routing problem focusing on performance-efficiency trade-offs. Given a user prompt and candidate LLMs, we aim to select the most cost-efficient model whose predicted response quality satisfies user-specified response quality tolerance.

### 2.1 LLM Routing Formulation

We formulate LLM routing as a constrained optimization problem. Denote the set of candidate LLMs as $\mathcal{C}$. Given a prompt $x_i$, each candidate model $c \in \mathcal{C}$ would generate a response $y_{i,c} = c(x_i)$ with quality $r_{i,c} = R(x_i, y_{i,c}) \in [0, 1]$ measured by a reward function capturing human preference alignment. The cost of invoking model $c$ is denoted by $v_{i,c}$.

To make the quality-cost trade-off explicit and controllable, we minimize cost subject to quality constraints. Users specify a quality tolerance $\tau \in [0, 1]$ defining acceptable quality degradation relative to the best available model. This induces a feasible set: $\mathcal{C}_\tau = \{c \in \mathcal{C} \mid G(r_{i,c}, \tau) \geq 0\}$, where $G$ is a performance-gating function determining whether candidate performance satisfies user tolerance. The optimal routing decision selects the most efficient model from the feasible set:

$$c_i^* = \arg\min_{c \in \mathcal{C}_\tau} v_{i,c}. \tag{1}$$

Since computing $r_{i,c}$ requires generating and evaluating responses, we train a quality estimator $\hat{r}_{i,c} = R_\theta(x_i, c)$ that predicts response quality using only the prompt and candidate identity:

$$\theta^* = \arg\min_\theta \sum_{i,c} \ell(R_\theta(x_i, c), r_{i,c}), \tag{2}$$

where $\ell$ is a regression loss and $r_{i,c}$ are ground-truth rewards from a calibrated reward model.

### 2.2 Routing Strategy

Given the quality estimates $\hat{r}_{i,c}$, we implement a quality-constrained, cost-optimal routing strategy through two stages: (1) filter out candidates not meeting quality tolerance, (2) select the most cost-efficient qualified model.

The gating function translates user tolerance $\tau$ to a quality threshold:

$$G(\hat{r}_{i,c}, \tau) = \hat{r}_{i,c} - r_{i,\text{th}} \geq 0, \tag{3}$$

$$\text{where} \quad r_{i,\text{th}} = \hat{r}_{i,\max} - \tau \cdot (\hat{r}_{i,\max} - \hat{r}_{i,\min}). \tag{4}$$

Here, $\hat{r}_{i,\min}$ is designed for min-max scaling when predictions have a non-zero baseline. We use dynamic per-prompt $\hat{r}_{i,\max} = \max_{c \in \mathcal{C}} \hat{r}_{i,c}$ to handle varying query complexity and fixed $\hat{r}_{i,\min} = 0$ for

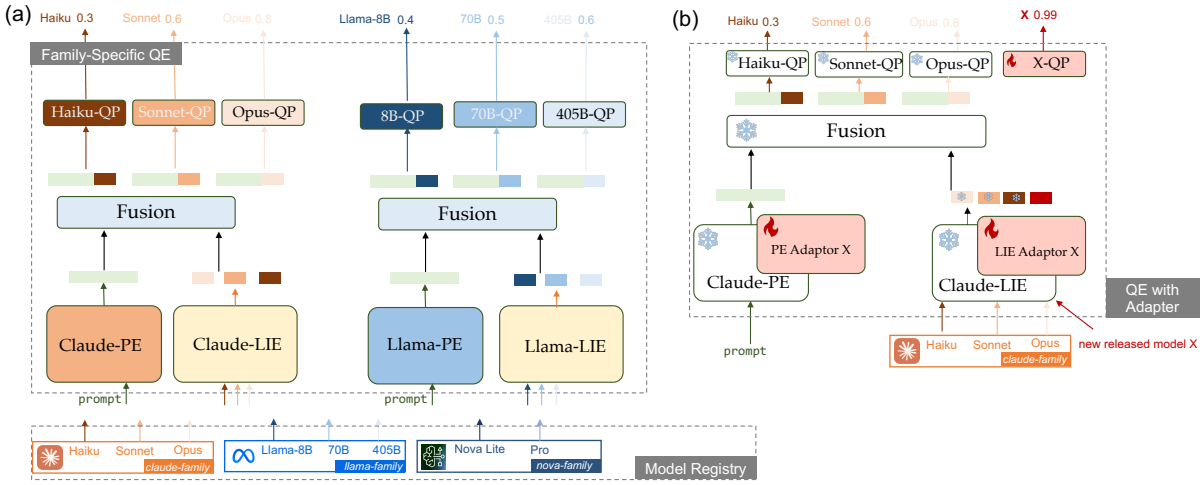


Figure 2: IPR Quality Estimator (QE) architecture. (a) Family-specific Quality Predictors (QP) process prompts with candidate model IDs to predict quality scores. (b) New models are integrated via lightweight adapters on frozen Prompt Encoder (PE) and LLM Identity Encoder (LIE) components.

stability. This hybrid strategy adapts to individual prompt difficulty while preventing threshold collapse (More routing strategies are discussed in Appendix H). Once the feasible set $\mathcal{C}_\tau(x_i)$ is determined, we select the minimum-cost candidate following Equation (1).

### 2.3 Evaluation

We evaluate IPR along two dimensions: quality prediction accuracy and end-to-end routing performance. For quality prediction, we use mean absolute error and ranking-based metrics, including Top-K accuracy and F1 scores. For routing performance, we introduce a area-under-the-curve (AUC) style metric named Bounded-ARQGC (and its variant, Relative-ARQGC) to measure quality-cost trade-offs across all tolerance settings and Cost Save Ratio (CSR) for different settings. Detailed metric definitions are provided in Appendix A.

## 3 Intelligent Prompt Routing

### 3.1 System Overview

Figure 1 illustrates the IPR platform, comprising three core components: (1) the *Quality Estimator* (QE) that predicts response quality for each candidate model, (2) the *Decision Optimization* (DO) module that executes quality-constrained routing decisions, and (3) the *Model Registry* that maintains model metadata and configurations.

The routing pipeline operates as follows: Upon receiving a user prompt with optional multi-turn context, the system captures the user's quality-cost preference through tolerance parameter $\tau \in [0, 1]$

**Algorithm 1** IPR Routing with User Tolerance

**Input:** Prompt $x$, candidate set $\mathcal{C}$, prices $\{v_c\}$, tolerance $\tau \in [0, 1]$, QE $R_\theta$, safety margin $\delta \geq 0$
**Output:** Routed model $c^\star$

$\mathbf{p} \leftarrow \mathrm{PE}(x)$ ▷ Prompt embedding (cached across turns if multi-turn)
**for** $c \in \mathcal{C}$ **do**
  $\mathbf{e}_c \leftarrow \mathrm{LIE}(c)$ ▷ Model identity embedding
  $\hat{r}_c \leftarrow \mathrm{QP}([\mathbf{p}; \mathbf{e}_c])$ ▷ Predicted quality (optionally calibrated)
**end for**
$\hat{r}_{\max} \leftarrow \max_{c \in \mathcal{C}} \hat{r}_c$
$r_{\mathrm{th}} \leftarrow (1 - \tau) \cdot \hat{r}_{\max}$ ▷ Per-prompt threshold with safety margin
$\mathcal{F} \leftarrow \{c \in \mathcal{C} \mid \hat{r}_c \geq r_{\mathrm{th}}\}$ ▷ Feasible candidates
**if** $\mathcal{F} = \emptyset$ **then**
  $\mathcal{F} \leftarrow \{\arg\max_c \hat{r}_c\}$ ▷ Fallback to predicted best
**end if**
$c^\star \leftarrow \arg\min_{c \in \mathcal{F}} v_c$ ▷ Minimize monetary cost; tie-break by $\hat{r}_c$
**return** $c^\star$

(where $\tau = 0$ enforces maximum quality and $\tau = 1$ maximizes cost savings). The Quality Estimator computes predicted quality scores $\hat{r}_{i,c}$ for each candidate $c \in \mathcal{C}$ using the learned estimator $R_\theta(x_i, c)$. These predictions feed into the Decision Optimization module, which applies tolerance-based filtering to identify feasible candidates and selects the minimum-cost model.

Our architecture achieves two critical objectives:

(1) *extensibility* through lightweight adapters for new models without full retraining, and (2) *efficiency* with fast routing decisions based on prompt embeddings. A sketch of the algorithm is shown in Algorithm 1.

### 3.2 Quality Estimator Architecture

The Quality Estimator predicts scalar reward scores approximating response quality for each prompt-model pair. As shown in Figure 2, it consists of three key components:

**(1) Prompt Encoder:** Maps input prompts to dense embeddings $\mathbf{p}_i = \text{PE}(x_i) \in \mathbb{R}^d$ capturing semantic features relevant for quality prediction. We employ family-specific encoders (e.g., Claude-PE, Llama-PE) to capture model-specific patterns.

**(2) LLM Identity Encoder:** Provides learnable embeddings $\mathbf{e}_c = \text{LIE}(c) \in \mathbb{R}^{d'}$ for each candidate model, encoding behavioral properties like verbosity and style.

**(3) Quality Predictor:** Fuses prompt and LLM embeddings via concatenation and predicts quality through a feed-forward network: $\hat{r}_{i,c} = \text{QP}(\text{Concat}(\mathbf{p}_i, \mathbf{e}_c))$.

We adopt family-specific architectures with separate prediction heads per model, enabling better within-family generalization and simplified integration of new models. Training uses **reward model scores** as supervision signals, providing fine-grained quality labels at scale (detailed in Appendix B).

IPR's modular design enables seamless integration of new LLMs without full retraining. When adding a new model, we freeze the core encoders and attach lightweight adapters that specialize the shared representations. This approach reduces integration and deployment time while preserving performance on existing models. Implementation details are provided in Appendix D.

## 4 Experiments

We train and evaluate IPR model on our dataset containing 1.5M prompts with quality annotations across multiple LLM families (details in Appendix G). This industrial-scale benchmark enables rigorous evaluation of routing systems under realistic conditions.

### 4.1 Dataset Collection

We construct the training and evaluation datasets using a diverse set of resources, covering open-domain dialogue, instruction-tuning, summarization, reasoning, and domain-specific question answering. The primary dataset includes responses from multiple language model families (Claude, Llama, Nova), where each instance contains outputs from all models within the same family, enabling direct comparison of response quality. Each response is annotated with a reward score assigned by the `Skywork/Skywork-Reward-Gemma-2-27B` model[1] (Liu et al., 2024a), which serves as the supervision signal for training the quality estimator. The scale and split of the Combined dataset across different model families are summarized in Table 1. More details are listed in Appendix G.

Table 1: IPR dataset size by model family and split. More details in Appendix G.

| Dataset | Subset | Claude | Llama | Nova |
|---|---|---|---|---|
| Combined | Training | 1,510,415 | 1,325,628 | 1,510,250 |
| | Dev | 5,641 | 4,976 | 5,640 |
| | Test | 5,642 | 5,032 | 5,641 |
| MS Marco | Test | 2,000 | 2,000 | 1,997 |
| Nvidia Chat | Test | 2,000 | 2,000 | 1,999 |

### 4.2 Experimental Setup

**Model Families and Candidates.** We evaluate IPR on three major LLM families, encompassing diverse model sizes and capabilities:

- **Claude family**: Claude-3-Haiku, Claude-3.5-Haiku, Claude-3.5-Sonnet variants
- **Llama family**: Llama-3.1-{8B, 70B}, Llama-3.2-{11B, 90B}, Llama-3.3-70B
- **Nova family**: Nova-Lite and Nova-Pro

These candidates represent different quality-cost trade-offs, allowing comprehensive routing evaluation. We chose to deploy family-specific routers due to their superior in-domain empirical performance (comparison of family-specific and unified router is shown in Appendix H).

**Baseline Methods.** We compare against: (1) **Static** routing to fixed models providing cost bounds, (2) **Random** uniform assignment, (3) **Oracle** routing with ground-truth quality scores, (4) **Budget-Aware Random** maintaining IPR's routing proportions but random assignment, and (5) **Classifier** following RouteLLM's approach.

[1] https://huggingface.co/Skywork/Skywork-Reward-Gemma-2-27B-v0.2

Table 2: Quality estimation performance on IPR test set. We report Mean Absolute Error (MAE), Top-1 Accuracy, and F1 scores for different router architectures. Best results are **bolded**, second best are underlined.

| Method | Claude | | | Llama | | | Nova | | |
|---|---|---|---|---|---|---|---|---|---|
| | MAE ↓ | Top-1 ↑ | F1-macro ↑ | MAE ↓ | Top-1 ↑ | F1-macro ↑ | MAE ↓ | Top-1 ↑ | F1-macro ↑ |
| IPR (RoBERTa-355M) | 0.09503 | 0.7025 | 0.6612 | 0.09283 | 0.7025 | 0.4825 | 0.09681 | 0.6500 | 0.5311 |
| IPR (Stella-400M) | 0.09478 | 0.7321 | 0.6629 | 0.08626 | 0.7154 | 0.5139 | 0.09597 | 0.6408 | 0.5828 |
| IPR (Qwen3-0.6B) | 0.09027 | 0.7353 | 0.6934 | 0.09120 | **0.7257** | 0.4940 | 0.09509 | 0.6642 | 0.5834 |
| IPR (Qwen3-4B) | 0.08540 | 0.7463 | 0.6857 | 0.08091 | 0.7237 | 0.5111 | 0.08384 | **0.6853** | 0.6016 |
| IPR (Qwen3-emb-0.6B) | 0.08988 | 0.7408 | **0.6982** | 0.08963 | 0.7217 | 0.4991 | 0.09279 | 0.6674 | 0.5717 |
| IPR (Qwen3-emb-4B) | **0.08390** | **0.7508** | 0.6931 | **0.07997** | 0.7094 | **0.5664** | **0.08281** | 0.6826 | **0.6070** |

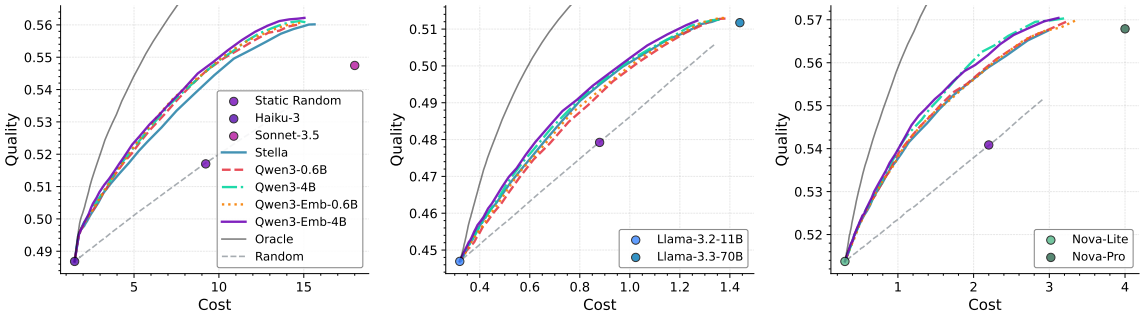


Figure 3: Quality and cost trade-offs under different user tolerance.

**Training Configuration.** Models are trained on IPR train set (1.5M samples) using 8 or 16 A100 GPUs.

### 4.3 Results

We evaluate IPR across three key dimensions: (1) quality estimation results, (2) overall routing performance across the full tolerance spectrum, and (3) cost savings at critical operating points. Our results demonstrate that for Claude family, IPR is able to achieve up to 43.9% cost reduction for Stella embedding model when maintaining quality equivalent to the most expensive model, while providing flexible quality-cost trade-offs for diverse user preferences.

**Quality Estimation Performance.** Table 2 presents quality estimation results on IPR test set across different backbone architectures. Scaling backbone size consistently improves prediction: Qwen3-emb-4B achieves lowest MAE (0.084 for Claude), 13.3% better than RoBERTa-355M. Embedding-based encoders outperform decoder counterparts at equivalent sizes. Notably, our selected Stella-400M achieves >73% top-1 accuracy while being 8× faster than 4B models.

**End-to-End Routing Performance.** Table 3 presents the overall routing performance across the full tolerance spectrum, measured by our primary metric Bounded-ARQGC. Across all model families, IPR variants significantly outperform baseline approaches, with the best configurations achieving 0.821 (Claude), 0.685 (Llama), and 0.766 (Nova) Bounded-ARQGC scores — representing relative improvements of 58.8%, 38.1%, and 57.6% over random routing respectively. The oracle router, which has access to ground-truth quality scores, establishes upper bounds of 0.915, 0.868, and 0.905, indicating room for future improvements in quality estimation.

Several key patterns emerge from these results. First, modest quality estimation improvements yield disproportionate routing gains: Stella-400M's 2-7% MAE reduction over RoBERTa translates to 3-21% higher Bounded-ARQGC. Figure 3 visualizes quality-cost trade-offs, showing IPR produces Pareto-optimal curves compared to baselines. Second, the relationship between model scale and routing effectiveness exhibits diminishing returns — Qwen3-emb-4B improves MAE by 8-11% over Stella-400M but yields only 2-16% better routing performance, suggesting that accurate relative quality rankings matter more than precise score predictions. Figure 3 visualizes quality-cost trade-offs, showing IPR produces Pareto-optimal curves dominating baseline approaches.

**Routing Latency and Efficiency.** IPR's routing decision requires a *single* forward pass of the

Table 3: Overall routing performance on IPR test set. We report Bounded-ARQGC (primary metric), Relative ARQGC improvement over random baseline. Best results (excluding Oracle) are **bolded**, second-best are underlined. Rows with gray shading indicate encoder-based architectures.

| Method | Claude | | Llama | | Nova | |
|---|---|---|---|---|---|---|
| | B-ARQGC ↑ | Rel-ARQGC ↑ | B-ARQGC ↑ | Rel-ARQGC ↑ | B-ARQGC ↑ | Rel-ARQGC ↑ |
| Oracle | 0.915 | 1.000 | 0.868 | 1.000 | 0.905 | 1.000 |
| Static (Strongest) | - | - | - | - | - | - |
| Static (Weakest) | - | - | - | - | - | - |
| Random | 0.517 | 0.434 | 0.496 | 0.504 | 0.486 | 0.431 |
| RouteLLM | 0.728 | 0.683 | 0.635 | 0.630 | 0.695 | 0.618 |
| IPR (Roberta-355M) | 0.732 | 0.695 | 0.628 | 0.625 | 0.707 | 0.622 |
| IPR (Stella-400M) | 0.799 | 0.724 | 0.663 | 0.676 | 0.731 | 0.650 |
| IPR (Qwen3-0.6B) | 0.808 | 0.730 | 0.641 | 0.653 | 0.739 | 0.658 |
| IPR (Qwen3-4B) | 0.813 | 0.743 | 0.672 | 0.685 | **0.766** | **0.687** |
| IPR (Qwen3-emb-0.6B) | 0.814 | 0.740 | 0.653 | 0.666 | 0.735 | 0.656 |
| IPR (Qwen3-emb-4B) | **0.821** | **0.756** | **0.685** | **0.698** | **0.766** | **0.687** |

Table 4: Router performance on IPR test dataset at 100% and 95% of the strongest model quality in Claude family.

| Method | 100% Quality | | | | 95% Quality | | | |
|---|---|---|---|---|---|---|---|---|
| | CSR | Acc. | Route Percentage | | CSR | Acc. | Route Percentage | |
| | | | Haiku | Sonnet | | | Haiku | Sonnet |
| oracle | 0.705 | 1.0 | 60.43 | 39.57 | 0.685 | 1.0 | 77.27 | 22.72 |
| RouteLLM | 0.425 | 0.605 | 50.28 | 49.72 | 0.712 | 0.732 | 75.32 | 24.68 |
| IPR(RoBERTa) | 0.385 | 0.638 | 48.8 | 51.2 | 0.658 | 0.756 | 79.2 | 10.8 |
| IPR(Stella) | 0.439 | 0.678 | 54.41 | 45.59 | 0.730 | 0.811 | 82.69 | 17.30 |
| IPR(Qwen3-0.6B) | **0.487** | 0.688 | **59.95** | 40.04 | 0.730 | 0.799 | 83.69 | 16.30 |
| IPR(Qwen3-4B) | 0.484 | **0.702** | 57.95 | 42.04 | 0.748 | **0.845** | 84.01 | 15.99 |
| IPR(Qwen3-Emb-0.6B) | 0.440 | 0.679 | 55.38 | 44.62 | 0.742 | 0.813 | **84.93** | 15.06 |
| IPR(Qwen3-Emb-4.B) | 0.465 | 0.695 | 56.10 | 43.89 | **0.754** | 0.843 | 84.25 | 15.74 |

prompt encoder to compute a prompt embedding, followed by tiny per-candidate MLP heads; *no autoregressive decoding* is involved. As a result, routing latency is *input-length dependent* but *output-length invariant*, and it adds only a few milliseconds before the selected endpoint is invoked. To make efficiency concrete, we benchmark on **1× A100-40GB (PCIe), CUDA 12.4** with batch=1, FP32, 100 warmup steps and 1,000 measured runs per setting. We vary input length (500 to 1000 tokens) and candidate set size ($|\mathcal{C}| = 5$ to 10). We report end-to-end wall-clock **P90/P99** (tokenization → encoder → heads → selection) and **peak memory** as in Table 5.

Table 5: Router latency and memory (end-to-end, batch=1) measured on a single A100-40GB GPU. Latency calculation includes tokenization, encoder forward, per-candidate heads, and selection.

| Name | Input (tok) | $\lvert\mathcal{C}\rvert$ | P90 (ms) | P99 (ms) | Mem (GB) |
|---|---|---|---|---|---|
| IPR (Stella) | 500 | 5 | 35.66 | 36.31 | 1.68 |
| IPR (Stella) | 1000 | 5 | 64.92 | 65.14 | 1.72 |
| IPR (Stella) | 1000 | 10 | 67.03 | 67.13 | 1.76 |
| IPR (Qwen3-0.6B) | 500 | 5 | 56.79 | 62.69 | 3.02 |
| IPR (Qwen3-0.6B) | 1000 | 5 | 115.30 | 116.74 | 3.80 |
| IPR (Qwen3-0.6B) | 1000 | 10 | 118.54 | 119.02 | 3.83 |
| IPR (Qwen3-4B) | 500 | 5 | 277.66 | 278.07 | 16.00 |
| IPR (Qwen3-4B) | 1000 | 5 | 557.05 | 557.42 | 16.81 |
| IPR (Qwen3-4B) | 1000 | 10 | 560.24 | 560.52 | 16.82 |

**Performance at Critical Operating Points.** While aggregate metrics capture overall routing effectiveness, practical deployment often focuses on specific quality-cost targets. Table 4 examines router performance at two critical operating points: maintaining 100% quality parity with the strongest model and accepting 5% quality degradation. At the 100% quality threshold — where users demand performance equivalent to always using the most capable model — IPR with Qwen3-0.6B achieves 48.7% cost savings by routing 59.9% of prompts to the more efficient Haiku model. This demonstrates that nearly 60% of real-world prompts do not require the most expensive model to achieve

optimal quality. The routing distribution reveals how different backbones assess prompt complexity. Smaller encoders (RoBERTa-355M) exhibit more conservative routing, sending only 48.8% to Haiku, while mid-sized models like Qwen3-0.6B achieve better prompt discrimination.

Results on IPR test dataset suggest IPR's effectiveness: achieving substantial cost reductions while maintaining quality, with flexible user control over trade-offs. Comprehensive ablation studies validating our design choices are provided in Appendix H. To verify routing quality, we conduct blind human annotation studies detailed in Appendix E.

## 5 Related Works

Here, we focus on existing prompt routing approaches, and defer benchmarks and evaluations to Appendix I.

In the literature, different model designs and corresponding training strategies have been proposed for the LLM routing problem (Lu et al., 2024; Ding et al., 2024; Ong et al., 2024; Sikeridis et al., 2024; Jitkrittum et al., 2025; Feng et al., 2025; Su et al., 2025; Chuang et al., 2025; Stripelis et al., 2024; Mei et al., 2025; Sakota et al., 2024; Chen et al., 2023, 2024; Jin et al., 2025; Ding et al., 2025; Sikeridis et al., 2025; Jitkrittum et al., 2025; Pan et al., 2025; Zhuang et al., 2024, *inter alia*). HybridLLM (Ding et al., 2024) employs a BERT-based encoder to optimize the cost-quality trade-off by routing "easy" queries to resource-efficient smaller models and "hard" queries to larger, more capable models. EmbedLLM (Zhuang et al., 2024) introduces a specialized encoder-decoder network for embedding LLM representations. RouteLLM (Ong et al., 2024) implements a dynamic routing mechanism that intelligently routes prompts between a stronger and weaker LLM through various methodologies: similarity-weighted ranking, matrix factorization, BERT-based classification, and Causal LLM classification. Zooter (Lu et al., 2024) also deploys reward model scores as the supervision signals and train the router with RankNet loss (Burges, 2010). Additionally, it leverages a tag-based label enhancement strategy to remove reward model noises. GraphRouter (Feng et al., 2025) formulates LLM selection as edge prediction problem in a graph based framework and fully utilizes the information in the training data by jointly modeling the query-model, query-query, and model-model relationship. OmniRouter (Mei et al., 2025) formulates the routing task as a constrained optimization problem and leverages a hybrid retrieval-augmented predictor to predict the capabilities and costs of LLMs. Different from aforementioned works that deploy clustering or train with teacher forcing, PickLLM (Sikeridis et al., 2024) proposes a reinforcement learning-based routing framework that optimizes a composite reward function incorporating latency, computational cost, and response quality. IPR deploys a conventional supervised learning approach and focus on scaling training data mixture for robust LLM routing.

## 6 Conclusions and Future Works

We introduce Intelligent Prompt Routing — a low latency solution to cost efficient prompt routing. We detail our scientific experimentation: including curation of a large-scale training and evaluation dataset, design of evaluation metrics Bounded-ARQGC and CSR, different model architecture, training strategy ablations throughout the product development. Our future works will include incorporating multifaceted evaluations and supporting new model releases on our platform.

## Limitations

While IPR demonstrates strong performance in production deployment, several limitations merit discussion. First, our quality estimation relies on reward model scores as supervision signals, which may not perfectly capture all aspects of human preference, particularly for specialized domains or creative tasks. Second, the current framework assumes independent routing decisions per prompt without considering conversation-level context or user session patterns, potentially missing optimization opportunities in multi-turn interactions. Third, our evaluation focuses on three model families (Claude, Llama, Nova) on a single platform; generalization to other model families or deployment environments requires further validation. Finally, the modular adaptation mechanism, while efficient, still requires access to labeled data for new models, which may not be immediately available upon model release. Addressing these limitations, particularly through online learning from user feedback and session-aware routing, represents important directions for future work.

## References


Christopher JC Burges. 2010. From ranknet to lambdarank to lambdamart: An overview. *Learning*, 11(23-581):81.

Iñigo Casanueva, Tadas Temcinas, Daniela Gerz, Matthew Henderson, and Ivan Vulic. 2020. Efficient intent detection with dual sentence encoders. In *Proceedings of the 2nd Workshop on NLP for ConvAI - ACL 2020*. Data available at https://github.com/PolyAI-LDN/task-specific-datasets.

Lingjiao Chen, Matei Zaharia, and James Zou. 2023. Frugalgpt: How to use large language models while reducing cost and improving performance. *arXiv preprint arXiv:2305.05176*.

Shuhao Chen, Weisen Jiang, Baijiong Lin, James Kwok, and Yu Zhang. 2024. Routerdc: Query-based router by dual contrastive learning for assembling large language models. *Advances in Neural Information Processing Systems*, 37:66305–66328.

Yu-Neng Chuang, Prathusha Kameswara Sarma, Parikshit Gopalan, John Boccio, Sara Bolouki, Xia Hu, and Helen Zhou. 2025. Learning to route LLMs with confidence tokens. In *Forty-second International Conference on Machine Learning*.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, and 1 others. 2021. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*.

Mike Conover, Matt Hayes, Ankit Mathur, Jianwei Xie, Jun Wan, Sam Shah, Ali Ghodsi, Patrick Wendell, Matei Zaharia, and Reynold Xin. 2023. Free dolly: Introducing the world's first truly open instruction-tuned llm.

Dujian Ding, Ankur Mallick, Chi Wang, Robert Sim, Subhabrata Mukherjee, Victor Rühle, Laks V. S. Lakshmanan, and Ahmed Hassan Awadallah. 2024. Hybrid LLM: Cost-efficient and quality-aware query routing. In *The Twelfth International Conference on Learning Representations*.

Dujian Ding, Ankur Mallick, Shaokun Zhang, Chi Wang, Daniel Madrigal, Mirian Del Carmen Hipolito Garcia, Menglin Xia, Laks V. S. Lakshmanan, Qingyun Wu, and Victor Rühle. 2025. BEST-route: Adaptive LLM routing with test-time optimal compute. In *Forty-second International Conference on Machine Learning*.

Alexander Fabbri, Xiaojian Wu, Srini Iyer, Haoran Li, and Mona Diab. 2022. AnswerSumm: A manually-curated dataset and pipeline for answer summarization. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2508–2520, Seattle, United States. Association for Computational Linguistics.

Tao Feng, Yanzhen Shen, and Jiaxuan You. 2025. Graphrouter: A graph-based router for LLM selections. In *The Thirteenth International Conference on Learning Representations*.

Mor Geva, Daniel Khashabi, Elad Segal, Tushar Khot, Dan Roth, and Jonathan Berant. 2021. Did aristotle use a laptop? a question answering benchmark with implicit reasoning strategies. *Transactions of the Association for Computational Linguistics*, 9:346–361.

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, Xiao Bi, and 1 others. 2025. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*.

Qitian Jason Hu, Jacob Bieker, Xiuyu Li, Nan Jiang, Benjamin Keigwin, Gaurav Ranganath, Kurt Keutzer, and Shriyash Kaustubh Upadhyay. 2024. Routerbench: A benchmark for multi-LLM routing system. In *Agentic Markets Workshop at ICML 2024*.

Zhongzhan Huang, Guoming Ling, Yupei Lin, Yandong Chen, Shanshan Zhong, Hefeng Wu, and Liang Lin. 2025. Routereval: A comprehensive benchmark for routing llms to explore model-level scaling up in llms. *arXiv preprint arXiv:2503.10657*.

Aaron Jaech, Adam Kalai, Adam Lerer, Adam Richardson, Ahmed El-Kishky, Aiden Low, Alec Helyar, Aleksander Madry, Alex Beutel, Alex Carney, and 1 others. 2024. Openai o1 system card. *arXiv preprint arXiv:2412.16720*.

Dongfu Jiang, Xiang Ren, and Bill Yuchen Lin. 2023. LLM-blender: Ensembling large language models with pairwise ranking and generative fusion. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 14165–14178, Toronto, Canada. Association for Computational Linguistics.

Ruihan Jin, Pengpeng Shao, Zhengqi Wen, Jinyang Wu, Mingkuan Feng, Shuai Zhang, and Jianhua Tao. 2025. Radialrouter: Structured representation for efficient and robust large language models routing. *arXiv preprint arXiv:2506.03880*.

Wittawat Jitkrittum, Harikrishna Narasimhan, Ankit Singh Rawat, Jeevesh Juneja, Zifeng Wang, Chen-Yu Lee, Pradeep Shenoy, Rina Panigrahy, Aditya Krishna Menon, and Sanjiv Kumar. 2025. Universal model routing for efficient llm inference. *arXiv preprint arXiv:2502.08773*.

Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph Gonzalez, Hao Zhang, and Ion Stoica. 2023. Efficient memory management for large language model serving with pagedattention. In *Proceedings of the 29th Symposium on Operating Systems Principles*, pages 611–626.

LAION-AI. 2023. Laion-ai open assistant. https://github.com/LAION-AI/Open-Assistant.

Chris Yuhao Liu, Liang Zeng, Jiacai Liu, Rui Yan, Jujie He, Chaojie Wang, Shuicheng Yan, Yang Liu, and Yahui Zhou. 2024a. Skywork-reward: Bag of tricks for reward modeling in llms. *arXiv preprint arXiv:2410.18451*.

Zihan Liu, Wei Ping, Rajarshi Roy, Peng Xu, Chankyu Lee, Mohammad Shoeybi, and Bryan Catanzaro. 2024b. Chatqa: Surpassing gpt-4 on conversational qa and rag. *arXiv preprint arXiv:2401.10225*.

Keming Lu, Hongyi Yuan, Runji Lin, Junyang Lin, Zheng Yuan, Chang Zhou, and Jingren Zhou. 2024. Routing to the expert: Efficient reward-guided ensemble of large language models. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 1964–1974, Mexico City, Mexico. Association for Computational Linguistics.

Kai Mei, Wujiang Xu, Shuhang Lin, and Yongfeng Zhang. 2025. Omnirouter: Budget and performance controllable multi-llm routing. *arXiv preprint arXiv:2502.20576*.

Tri Nguyen, Mir Rosenberg, Xia Song, Jianfeng Gao, Saurabh Tiwary, Rangan Majumder, and Li Deng. 2016. MS MARCO: A human generated machine reading comprehension dataset. *CoRR*, abs/1611.09268.

Isaac Ong, Amjad Almahairi, Vincent Wu, Wei-Lin Chiang, Tianhao Wu, Joseph E Gonzalez, M Waleed Kadous, and Ion Stoica. 2024. Routellm: Learning to route llms from preference data. In *The Thirteenth International Conference on Learning Representations*.

Zhihong Pan, Kai Zhang, Yuze Zhao, and Yupeng Han. 2025. Route to reason: Adaptive routing for llm and reasoning strategy selection. *arXiv preprint arXiv:2505.19435*.

Marija Sakota, Maxime Peyrard, and Robert West. 2024. Fly-swat or cannon? cost-effective language model choice via meta-modeling. In *Proceedings of the 17th ACM International Conference on Web Search and Data Mining*, WSDM '24, page 606–615, New York, NY, USA. Association for Computing Machinery.

Dimitrios Sikeridis, Dennis Ramdass, and Pranay Pareek. 2024. Pickllm: Context-aware rl-assisted large language model routing. *arXiv preprint arXiv:2412.12170*.

Dimitrios Sikeridis, Dennis Ramdass, and Pranay Pareek. 2025. Pickllm: Context-aware rl-assisted large language model routing. In *International Workshop on AI for Transportation*, pages 227–239. Springer.

Dimitris Stripelis, Zijian Hu, Jipeng Zhang, Zhaozhuo Xu, Alay Dilipbhai Shah, Han Jin, Yuhang Yao, Salman Avestimehr, and Chaoyang He. 2024. Tensoropera router: A multi-model router for efficient llm inference. *arXiv preprint arXiv:2408.12320*.

Jiayuan Su, Fulin Lin, Zhaopeng Feng, Han Zheng, Teng Wang, Zhenyu Xiao, Xinlong Zhao, Zuozhu Liu, Lu Cheng, and Hongwei Wang. 2025. Cp-router: An uncertainty-aware router between llm and lrm. *arXiv preprint arXiv:2505.19970*.

Alon Talmor, Jonathan Herzig, Nicholas Lourie, and Jonathan Berant. 2019. CommonsenseQA: A question answering challenge targeting commonsense knowledge. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4149–4158, Minneapolis, Minnesota. Association for Computational Linguistics.

Rohan Taori, Ishaan Gulrajani, Tianyi Zhang, Yann Dubois, Xuechen Li, Carlos Guestrin, Percy Liang, and Tatsunori B. Hashimoto. 2023. Stanford alpaca: An instruction-following llama model. https://github.com/tatsu-lab/stanford_alpaca.

Guan Wang, Sijie Cheng, Xianyuan Zhan, Xiangang Li, Sen Song, and Yang Liu. 2024. Openchat: Advancing open-source language models with mixed-quality data. In *The Twelfth International Conference on Learning Representations*.

Weizhe Yuan, Graham Neubig, and Pengfei Liu. 2021. Bartscore: Evaluating generated text as text generation. *Advances in neural information processing systems*, 34:27263–27277.

Murong Yue, Jie Zhao, Min Zhang, Liang Du, and Ziyu Yao. 2024. Large language model cascades with mixture of thought representations for cost-efficient reasoning. In *The Twelfth International Conference on Learning Representations*.

Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. 2019. HellaSwag: Can a machine really finish your sentence? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4791–4800, Florence, Italy. Association for Computational Linguistics.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Tianle Li, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zhuohan Li, Zi Lin, Eric Xing, Joseph E. Gonzalez, Ion Stoica, and Hao Zhang. 2024a. LMSYS-chat-1m: A large-scale real-world LLM conversation dataset. In *The Twelfth International Conference on Learning Representations*.

Lianmin Zheng, Liangsheng Yin, Zhiqiang Xie, Chuyue Sun, Jeff Huang, Cody Hao Yu, Shiyi Cao, Christos Kozyrakis, Ion Stoica, Joseph E. Gonzalez, Clark Barrett, and Ying Sheng. 2024b. SGLang: Efficient execution of structured language model programs. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*.

Banghua Zhu, Evan Frick, Tianhao Wu, Hanlin Zhu, and Jiantao Jiao. 2023. Starling-7b: Improving llm helpfulness & harmlessness with rlaif.

Richard Zhuang, Tianhao Wu, Zhaojin Wen, Andrew Li, Jiantao Jiao, and Kannan Ramchandran. 2024. Embedllm: Learning compact representations of large language models. *arXiv preprint arXiv:2410.02223*.

# A Evaluation Metrics

## A.1 Quality Prediction Metrics

Since routing decisions depend on accurate quality predictions, we validate the estimator's ranking ability:

**Top-K Accuracy.** Measures whether the predicted top-$k$ models match the ground-truth top-$k$ models in exact order. For $N$ candidates, we report accuracies for $k \in \{1, ..., N-1\}$.

**Top-K F1.** Relaxes the ordering constraint by measuring set overlap between predicted and ground-truth top-$k$ models, providing a more forgiving assessment of ranking quality.

## A.2 Routing Performance Metrics

**Bounded-ARQGC.** To evaluate routing quality across varying cost-quality trade-offs, we introduce *Bounded Average Response Quality Gain under Cost* (Bounded-ARQGC). This metric generalizes the area under the quality-cost curve, normalized to $[0, 1]$.

Formally, let $Q(\alpha)$ denote the average response quality achieved when the router operates at cost budget $\alpha \cdot C_{\text{max}}$, where $C_{\text{max}}$ is the cost of always using the most expensive model. Bounded-ARQGC is defined as:

$$\text{Bounded-ARQGC} = \int_0^1 \frac{Q(\alpha) - Q_{\text{min}}}{Q_{\text{max}} - Q_{\text{min}}} d\alpha, \quad (5)$$

where $Q_{\text{min}}$ and $Q_{\text{max}}$ are the qualities achieved by always using the cheapest and best models respectively. Notably, Bounded-ARQGC has following key properties:

- Random routing yields $\approx 0.5$ (diagonal line).
- Perfect routing approaches $1.0$ (upper-left corner).
- Higher values indicate better cost-quality trade-offs.

Different from metrics that evaluate at fixed operating points or quality threshold values, Bounded-ARQGC captures routing performance across all possible tolerance settings, making it ideal for comparing routers without committing to specific deployment configurations.

**Cost Save Ratio (CSR).** For practical deployment decisions, we report cost savings at specific quality targets:

$$\text{CSR}(\tau) = \frac{v_{\text{best}} - v_{\text{router}}(\tau)}{v_{\text{best}}}, \quad (6)$$

where $v_{\text{router}}(q)$ is the cost to achieve quality level $q$ relative to the best model's quality. For instance, CSR(100%) indicates cost savings while maintaining the best model's full quality—our primary operating point in production.

# B Reward Modeling for Quality Supervision

Training an accurate quality estimator requires large-scale supervision signals that capture human preferences over model responses. While human annotations provide the gold standard, their cost prohibits scaling to the millions of prompts needed for robust routing. We address this challenge by leveraging reward models (RMs) as automated quality evaluators.

Our approach treats response quality estimation as a regression problem: given a prompt $x_i$ and candidate model $c$'s response $y_{i,c}$, the reward model produces a quality score $r_{i,c} = \text{RM}(x_i, y_{i,c}) \in [0, 1]$. The quality estimator then learns to predict these scores directly from prompts without generating responses: $\hat{r}_{i,c} = R_\theta(x_i, c)$.

This formulation provides three key advantages:

**Fine-grained supervision**: Unlike binary preferences or categorical labels, continuous RM scores capture subtle quality differences between models. For instance, while models may produce acceptable responses for simple queries, RMs can distinguish the incrementally better coherence or completeness that justifies routing to more capable models.

**Alignment with human judgment**: We validate that RM-based rankings align with human preferences through systematic evaluation. Model orderings derived from RM scores (e.g., Claude-3.5-Sonnet > Claude-3-Opus > Claude-3-Haiku) match human annotator rankings with 85% agreement, significantly outperforming LLM-as-a-Judge approaches.

**Distribution properties**: RM scores exhibit favorable statistical properties for learning, with well-separated score distributions across models (typical separation of 0.1-0.2 between adjacent models) compared to the compressed ranges produced by LLM judges. This separation provides clearer learning signals and more stable gradient updates during training.

In practice, we employ the Skywork-Reward model (Liu et al., 2024a) to generate training labels, chosen for its strong correlation with human preferences and computational efficiency. This approach enables us to create training datasets of over 1.5M examples while maintaining quality comparable to human-annotated data.

# C Quality Estimator Implementation Details

## C.1 Architectural Specifications

**Prompt Encoder Details:** The prompt encoder uses a pretrained transformer model with fixed architecture, fine-tuned on paired prompt-score examples. For family-specific quality estimation, each model family maintains independent prompt encoders initialized from the same base encoder. Typical embedding dimension $d = 768$ for efficiency.

**LLM Identity Encoder Details:** Learnable embeddings for each candidate model with dimension $d' = 128$. These embeddings are learned jointly with the predictor and capture model-specific behavioral patterns. For modular extension, we maintain separate LLM Identity Encoders per model family.

**Fusion Module Architecture:** The concatenated embeddings pass through a 2-layer feed-forward network with ReLU activation:

$$\mathbf{z}_{i,c} = \text{Concat}(\mathbf{p}_i, \mathbf{e}_c) \tag{7}$$

$$\mathbf{h} = \text{ReLU}(\mathbf{W}_1 \mathbf{z}_{i,c} + \mathbf{b}_1) \tag{8}$$

$$\hat{r}_{i,c} = \sigma(\mathbf{W}_2 \mathbf{h} + \mathbf{b}_2) \tag{9}$$

where $\sigma$ is the sigmoid function to ensure output in $[0, 1]$.

## C.2 Unified vs. Family-Specific Design

While a unified QE architecture with shared encoders and single prediction head is more compact, our experiments show superior performance using family-specific variants: - 5-8% higher ranking accuracy within families - Better generalization to new models within the same family - Simplified debugging and model-specific optimization - Reduced interference between models with distinct output behaviors

# D Modular Adaptation Implementation

To ensure extensibility, our design incorporates lightweight adapter modules for seamless integration of new LLMs. As illustrated in Figure 2, we freeze core encoders after initial training and attach learnable adapters for new models.

**Adapter Architecture:** - **PE Adapter X**: 2-layer feed-forward network with residual connection, inserted after frozen prompt encoder - **LIE Adapter X**: Single linear transformation after frozen identity encoder - **New QP Head**: Model-specific prediction head trained from scratch

**Training Procedure:** 1. Freeze all existing model components 2. Initialize adapters with identity mapping 3. Train only adapters and new QP head on data mixture: - 70% new model data - 30% existing model data (for consistency) 4. Use consistency loss to maintain performance:

$$\mathcal{L} = \mathcal{L}_{\text{new}} + \lambda \sum_{i,c \in \mathcal{C}_{\text{old}}} ||\hat{r}_{i,c} - \hat{r}_{i,c}^{\text{frozen}}||^2 \tag{10}$$

This framework reduces new model integration from 2-3 days of full training to 3-4 hours of adapter training, while maintaining 98%+ performance on existing models.

# E Human Annotation Results

We conducted a comprehensive evaluation of IPR-selected responses through human annotations following the specified protocol. Our evaluation framework employed a multi-batch annotation strategy to ensure robust and reliable assessments across different model families.

The human evaluation dataset was derived from a subset of the IPR test dataset. We deliberately excluded coding-related tasks from the evaluation due to limitations in annotation expertise for technical code assessment. The resulting dataset comprised **895 prompts**, each evaluated across **9 different models**, including 4 models from the Claude family and 5 models from the Llama family, resulting in 8055 responses.

We employed a rigorous evaluation protocol where each response underwent three blind annotation passes. The final scores were determined through majority voting across these passes, followed by calculation of the average overall satisfaction score for each model.

**Overall Satisfaction Scores** The human annotation results revealed clear performance hierarchies within both model families. Table 6 presents the average overall satisfaction scores after majority voting.

Table 6: Average Overall Satisfaction Scores by Model.

| Model | Average Score |
|---|---|
| *Claude Family* | |
| Claude 3 Haiku | 0.8209 |
| Claude 3.5 Sonnet V1 | 0.8220 |
| Claude 3.5 Haiku | 0.8654 |
| Claude 3.5 Sonnet V2 | **0.8708** |
| *Llama Family* | |
| Llama 3.1 8B | 0.7901 |
| Llama 3.1 70B | 0.8136 |
| Llama 3.2 11B | 0.8554 |
| Llama 3.2 90B | 0.8692 |
| Llama 3.3 70B | **0.8767** |

To provide more granular insights into model performance differences, we conducted pairwise comparisons for priority model pairs. Table 7 presents the win-tie-lose rates for three key comparisons that are critical for routing decisions.

Table 7: Pairwise LLM Comparison Results

| Pair | Win (%) | Tie (%) | Lose (%) |
|---|---|---|---|
| Haiku-3 vs. Sonnet 3.5 | 11.28 | 52.85 | 31.73 |
| Haiku-3.5 vs. Sonnet 3.5 | 14.19 | 61.68 | 16.54 |
| Llama-3.2 11B vs. 3.3-70B | 12.74 | 53.18 | 20.11 |

The human annotation results demonstrate strong alignment with our expected performance hierarchies for IPR decisions. Specifically, we observe the following orderings for all priority model pairs:

1. **Claude Family**: Haiku $<$ Sonnet 3.5 V2 and Haiku 3.5 $<$ Sonnet 3.5 V2
2. **Llama Family**: Llama 3.2 11B $<$ Llama 3.3 70B

These rankings are consistent with the reward model score comparisons, providing convergent validity for our evaluation framework. The high percentage of ties in pairwise comparisons (ranging from 52.85% to 61.68%) suggests that model capabilities overlap significantly for many tasks, highlighting the importance of careful model selection based on specific use case requirements.

# F Cost Calculation Formula and Detailed Model Costs

We compute the routing cost as the sum of both input and output token cost per 1M tokens based on the Amazon Bedrock price list as of March 19, 2025. In our following formula, we use normalized cost to make it invariant to different datasets with different prompt or response lengths.

Specifically, given:

- A prompt $x_i$
- The selected LLM is $m_i$
- The input cost per token is $P_{m_i}$ for LLM $m_i$
- The output cost per token is $Q_{m_i}$ for LLM $m_i$
- The input prompt length is $L_{x_i}$
- The output response length for prompt $x_i$ and model $m_i$ is $O(x_i, m_i)$

The normalized cost for $N$ prompts and their correspondingly selected LLMs is computed as:

$$C = \frac{\sum_i^N L_{x_i} \times P_{m_i}}{\sum_i^N L_{x_i}} + \frac{\sum_i^N O(x_i, m_i) \times Q_{m_i}}{\sum_i^N O(x_i, m_i)} \tag{11}$$

## F.1 Language Model Unit Prices

Model prices for each LLM candidate is listed in Table Table 8. **Note:** Prices are subject to change.

Table 8: Model pricing per 1,000 Tokens

| LLM Family | Model | Input Tokens | Output Tokens |
|---|---|---|---|
| **Anthropic** | Claude 3.5 Sonnet V2 | $0.003 | $0.015 |
| | Claude 3.5 Sonnet V1 | $0.003 | $0.015 |
| | Claude 3.5 Haiku | $0.0008 | $0.004 |
| | Claude 3 Haiku | $0.00025 | $0.00125 |
| **Llama** | Llama 3.3 Instruct (70B) | $0.00072 | $0.00072 |
| | Llama 3.2 Instruct (90B) | $0.00072 | $0.00072 |
| | Llama 3.2 Instruct (11B) | $0.00016 | $0.00016 |
| | Llama 3.1 Instruct (70B) | $0.00099 | $0.00099 |
| | Llama 3.1 Instruct (8B) | $0.00022 | $0.00022 |
| **Nova** | Nova Pro | $0.0008 | $0.0032 |
| | Nova Lite | $0.00006 | $0.00024 |

# G Dataset Collection

The composition of the Combined training set is summarized in Table 9: the largest portion comes from a multi-turn chat corpus (approximately 61%), followed by instruction-tuning and knowledge-intensive datasets. This mixture provides broad

coverage across natural language task types, allowing the quality estimator to generalize effectively across diverse prompt styles. We determined the specific proportions in Table 9 by computing a weighted ratio for each constituent dataset, where the weight corresponds to the ratio of the original dataset size to the cumulative size of all original datasets. We then uniformly sampled from each dataset according to its assigned ratio, for instance, a dataset with proportion 60% contributed 60% of its datapoints to the combined training set.

The training set comprises approximately 1.5 million examples for Claude, with similar sizes for Llama and Nova after filtering out examples with response generation failures due to throttling or timeout. Development and test sets contain between 5,000 and 6,000 examples per model family and follow a similar prompt distribution.

To evaluate generalization, we include two held-out test sets: **MS Marco** (Nguyen et al., 2016) and **Nvidia Chat** (Liu et al., 2024b) which focus on retrieval-augmented question answering, each with around 2,000 prompts (uniformly sampled). All test responses are also scored by the `Skywork/Skywork-Reward-Gemma-2-27B` reward model to support evaluation.

Table 9: Training dataset composition by source dataset.

| Dataset Name | Count | Proportion |
|---|---|---|
| LMSYS-Chat-1M (Zheng et al., 2024a) | 925,278 | 61.26% |
| ShareGPT-Vicuna (Wang et al., 2024) | 201,922 | 13.37% |
| MixInstruct (Jiang et al., 2023) | 98,473 | 6.52% |
| Nectar (Zhu et al., 2023) | 98,177 | 6.50% |
| AnswerSumm (Fabbri et al., 2022) | 42,454 | 2.81% |
| HellaSwag (Zellers et al., 2019) | 41,801 | 2.77% |
| StrategyQA (Geva et al., 2021) | 39,385 | 2.61% |
| CommonsenseQA (Talmor et al., 2019) | 39,081 | 2.59% |
| BANKING77 (Casanueva et al., 2020) | 14,073 | 0.93% |
| GSM8K (Cobbe et al., 2021) | 9,771 | 0.65% |

## H Ablation Studies

We conduct comprehensive ablations to validate our design choices across three critical dimensions: training objectives, architectural decisions, and routing strategies.

Table 10: Comparison of training loss functions (averaged over three model families).

| Loss | B-ARQGC | Quality | CSR | Route Acc |
|---|---|---|---|---|
| MSE | 0.7361 | 0.5451 | 0.3130 | 0.6353 |
| Hinge Loss | 0.6897 | 0.5438 | 0.2660 | 0.6035 |
| ListNet | 0.7292 | 0.5448 | 0.2656 | 0.5673 |

**Training Loss Functions.** Table 10 compares different loss functions for training the quality estimator, averaged across all model families. While we experimented with ranking-based losses that directly optimize for relative ordering, MSE loss achieves the best overall performance with 0.736 Bounded-ARQGC, outperforming hinge loss by 6.7% and ListNet by 0.9%. This result can be explained by two factors. First, continuous regression targets provide richer gradient signals than pairwise or listwise comparisons, enabling more stable optimization. Second, MSE loss naturally captures the magnitude of quality differences, which proves crucial for threshold-based routing decisions. Interestingly, while hinge loss achieves comparable routing accuracy (60.3% vs 63.5%), it significantly underperforms in cost savings (26.6% vs 31.3%), suggesting that accurate quality magnitude estimation is more important than perfect ranking for cost-optimal routing.

**Family-Specific vs. Unified Routing.** Table 11 examines the trade-offs between training separate routers for each model family versus a single unified router. Family-specific routers consistently outperform unified approaches on in-domain data, achieving higher Bounded-ARQGC scores (0.799 vs 0.792 for Claude, 0.663 vs 0.659 for Llama, and 0.731 vs 0.729 for Nova). This specialization advantage stems from the reduced learning complexity — each router only needs to focus on quality patterns within a homogeneous model group. Conversely, unified routers excel at generalization: on out-of-distribution datasets, they achieve 5.7%, 1.6%, and 7.6% higher Bounded-ARQGC for Claude, Llama, and Nova respectively. This reveals a bias-variance trade-off where family-specific routers precisely capture in-domain patterns but may overfit to family-specific characteristics. Given our production emphasis on in-domain performance, we deploy family-specific routing while recognizing unified routing's merits for heterogeneous workloads.

**Routing Strategy and Threshold Calibration.** We ablate two key aspects of our routing algorithm: the threshold computation method (dynamic vs. static) and the quality reference point. As described in Section 2.2, dynamic thresholds adapt to each prompt's quality distribution while static thresholds use global statistics. As shown in Figure 6, our experiments reveal that Dynamic Max and Dynamic MinMax achieves the optimal AUC

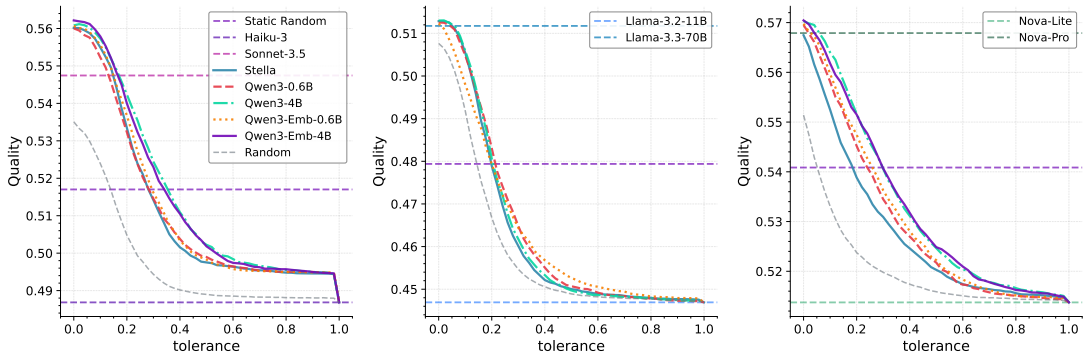


Figure 4: Quality v.s. tolerance with different QE backbones.

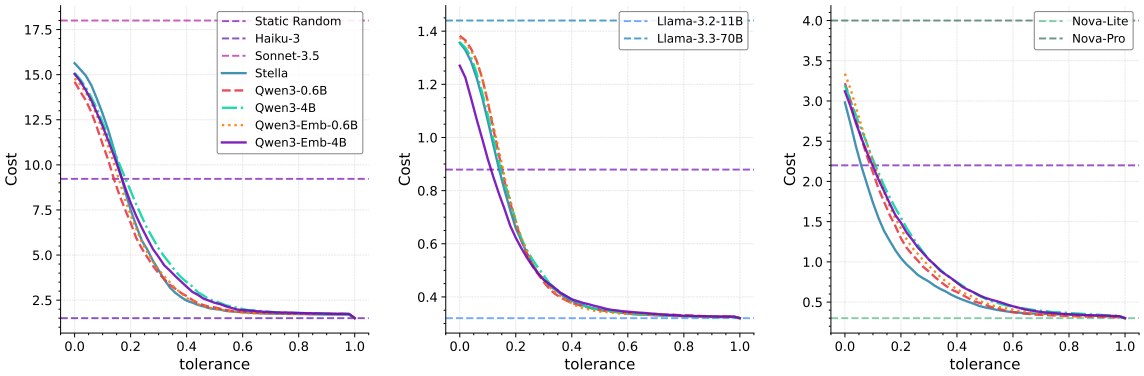


Figure 5: Cost v.s. tolerance with different QE backbones.

compares to others. Among this two, Dynamic Max has more smooth quality and cost curve vs tolerance compared to Dynamic MinMax, giving user more freedom to control the routing behavior. This hybrid strategy effectively handles prompts with varying difficulty — easy prompts with high quality scores across all models benefit from dynamic adaptation, while the fixed minimum prevents threshold collapse for uniformly challenging prompts. This also indicates the per-prompt normalization is crucial: without adapting thresholds to individual quality distributions, routers exhibit excessive conservatism, routing more prompts to expensive models unnecessarily.

## I Extended Discussions on Related Works

**Routing benchmarks.** Existing LLM routing benchmarks are mostly curated from popular NLP datasets covering different facets of LLM usage. MixInstruct (Jiang et al., 2023) consists of 110k examples focusing on the chat capability of LLMs. The mixture is primarily from four sources: Alpaca-GPT4 (Taori et al., 2023), Dolly-15K (Conover et al., 2023), GPT4All-LAION (LAION-AI, 2023) and ShareGPT (Wang et al., 2024). RouterBench (Hu et al., 2024) constructs a benchmark with over 405k inference outcomes from 11 representative LLMs across 8 diverse datasets to support the development of routing strategies. Routing strategies covered in RouterBench are simple methods like KNN and MLP routers. RouterEval (Huang et al., 2025) is a concurrent work that curates a large scale evaluation benchmark, spanning 12 popular LLM evaluations across various areas such as commonsense reasoning, semantic understanding, etc, and including over 200M performance records. This technical report describes our solution to curate IPR dataset, an industrial-scale LLM routing benchmark that focuses on natural language understanding and text generation capabilities of LLMs, and includes models currently served on our platform.

**Routing evaluations.** Existing works (Ong et al., 2024; Hu et al., 2024; Huang et al., 2025; Lu et al., 2024, *inter alia*) mostly categorize evaluation metrics into two groups: (1) effectiveness metrics and (2) efficiency metrics.[2]

Effectiveness metrics directly evaluate measure whether a query is routed to the most performant routing candidates. RouterEval (Huang et al., 2025), CP-Router (Su et al., 2025) and Self-

[2]Some works like (Huang et al., 2025) also refer to as (1) routing performance metrics and (2) cost reduction metrics.

Table 11: In- and out-of-distribution performance comparison of family-specific and unified routers. Cost-saving ratio (CSR) and routing accuracy (ACC) are reported at 100% best candidate performance. Bold values indicate superior performance within each distribution type.

| | | In-Distribution | | | | Out-of-Distribution | | | |
|---|---|---|---|---|---|---|---|---|---|
| **Model** | **Type** | **MAE ↓** | **B-ARQGC ↑** | **CSR ↑** | **ACC ↑** | **MAE ↓** | **B-ARQGC ↑** | **CSR ↑** | **ACC ↑** |
| Claude | specific | 0.09478 | **0.799** | **0.439** | **0.678** | 0.1532 | 0.523 | 0.369 | 0.57 |
| | unified | 0.1005 | 0.792 | 0.421 | 0.668 | **0.142** | **0.553** | **0.398** | **0.61** |
| Llama | specific | **0.08626** | **0.663** | **0.0773** | **0.677** | 0.1221 | 0.512 | 0.0712 | 0.59 |
| | unified | 0.08710 | 0.659 | 0.0720 | 0.672 | **0.1190** | **0.520** | **0.0725** | **0.60** |
| Nova | specific | **0.09597** | **0.731** | **0.255** | **0.652** | 0.1447 | 0.525 | 0.152 | 0.60 |
| | unified | 0.1021 | 0.729 | 0.242 | 0.648 | **0.1324** | **0.565** | **0.180** | **0.64** |

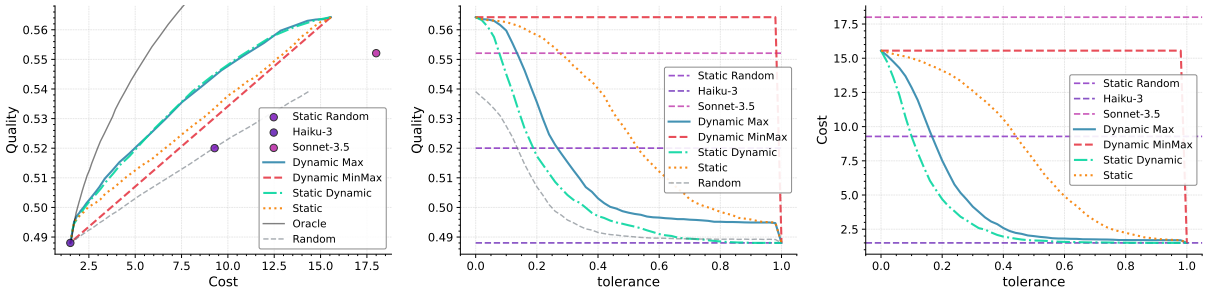


Figure 6: Quality-performance trade-off (left), quality-tolerance (middle), and cost-tolerance(right) relationship with different routing strategies

Table 12: Routing strategy comparison.

| **Strategy** | **min** | **max** |
|---|---|---|
| dynamic max | 0 | dynamic |
| dynamic minmax | dynamic | dynamic |
| static dynamic | static | dynamic |
| static | static | static |

REF (Chuang et al., 2025) evaluate routing effectiveness by Accuracy, i.e., the correctness of final predictions, which can be considered a top-1 metric. HybridLLM (Ding et al., 2024) focus on text generation tasks and adopt BARTScore (Yuan et al., 2021) as the quality/effectiveness metric. RouteLLM (Ong et al., 2024) defines an *average response quality* score that covers different NLP tasks, e.g., correctness on golden-labeled dataset or a numerical rating. In this technical report, we focus on top-1 accuracy, F1 scores as well as AUC as the main performance metrics.

Different from straightforward effectiveness metrics, there lacks a established efficiency metric that applies to different models and platforms, due to different notions of cost definitions. For example, HybridLLM (Ding et al., 2024) directly use the monetary cost as a proxy for the cost metric, e.g., $ per 1M tokens. Some works like (Su et al., 2025) uses number of tokens to represent the cost. In contrast to this absolute cost metric, works such as RouteLLM (Ong et al., 2024) adopt relative cost efficiency metric. For example, RouteLLM (Ong et al., 2024) define the cost efficiency metric as the percentage of calls to strong models. For evaluation of IPR, we adopt the proposed Bounded Average Response Quality Gain under Cost and Cost Save Ratio as the main efficiency metrics. We should note that, due to quick advancement of LLM inference optimization, exemplified by frameworks like vLLM (Kwon et al., 2023) and SGLang (Zheng et al., 2024b), the cost metric needs to be actively refreshed to reflect the actual inference cost.

The central goal of the LLM routing problem is to optimize the trade-off between effectiveness and efficiency. Various metrics have been used for evaluation and subsequently adopted as training objectives for the Router. We skip the detailed discussions and kindly refer readers to those original works for design rationales and exact formulations.