Flexible Routing via Uncertainty Decomposition

Charlotte Peale [*1†] **Siddartha Devic** [2†]
Parikshit Gopalan [3] **Udi Wieder** [3] **Aravind Gollakota** [3]

Abstract

A key strategy for balancing performance and cost in modern machine learning systems is to dynamically route queries to either a low-cost model or a more expensive oracle (such as a large pretrained model or human expert), an approach known as model routing. In this work we present a new uncertainty-aware router that (1) avoids unnecessary oracle calls on inherently ambiguous queries, and (2) adapts dynamically to different loss functions and cost parameters through simple hyperparameter changes, without retraining. Our method, applicable to any classification setting where multiple independent annotations per input are available, is based on decomposing total uncertainty into irreducible and reducible components using higher-order predictors (Ahdritz et al., 2025). This enables a unified approach to both routing and abstention: predict with the weak model when uncertainty is low, route to the oracle when reducible uncertainty is high, and abstain when irreducible uncertainty is high. Our router comes with strong theoretical guarantees bounding regret relative to optimal task-specific routers. We conduct experiments on both synthetic and real-world datasets that demonstrate the benefits of our approach in suitable regimes—in particular, whenever reducible and irreducible uncertainty are not too correlated.

1. Introduction

A core challenge in modern machine learning is navigating the trade-off between model performance and computational cost. As models have grown in scale and capability, they have also become more expensive to query in terms of cost, latency, and other resources. In order to build efficient and modular systems, it is increasingly important to be able to dynamically select the most appropriate model for a particular query or task, a task known as *model routing*. Routing seeks to optimize this trade-off by handling "simple" queries with a low-cost model (e.g., a small local model or non-thinking LLM) and deferring "complex" queries to an expensive, high-quality oracle (e.g., a powerful server-side model, ChatGPT 5's "Thinking" mode, or human experts in automated data annotation pipelines).

There has been a large body of work on routing and closely related problems such as orchestration, selective prediction, and learning to defer. Most existing methods can be grouped into two main categories: **Confidence-based methods**, which leverage uncertainty estimates to route to the oracle (or alternatively abstaining) when the weak model's confidence is low (see e.g. (Geifman & El-Yaniv, 2017)), and **Supervision-based methods**, which use an auxiliary router trained directly to predict when the stronger model will out-perform the weaker model (see e.g. (Strong et al., 2025; Ong et al., 2025)). While both have their strengths, they also have key weaknesses. Confidence-based methods often fail to account for irreducible (aleatoric) uncertainty (such as ambiguous or noisy utterances), needlessly routing to the oracle even when it would also perform poorly. Conversely, supervision-based routers require expensive, task-specific training and lack the flexibility to adapt to changing API costs or downstream losses without retraining. See Section A for an extended discussion of related work.

Our contribution We present a *flexible, uncertainty-aware router* that integrates both routing and abstention into a unified framework. Our approach builds on the recently introduced abstraction of a higher-order predictor (Ahdritz et al., 2025), which predicts mixtures over label distributions.[1] Assuming a property known as higher-order calibration (see Definition 3.1), we can decompose the expected loss on an input into irreducible (aleatoric) and reducible (epistemic) components. By leveraging this mean-

[*]Lead author. [†]Work done while interning at Apple. [1]Department of Computer Science, Stanford University [2]Department of Computer Science, University of Southern California [3]Apple. Correspondence to: Charlotte Peale <cpeale@stanford.edu>.

Accepted to the ICML 2026 Workshop on Statistical Frameworks for Uncertainty in Agentic Systems, Seoul, South Korea, 2026.
Copyright 2026 by the author(s).

[1]We point out that higher-order predictors also arise naturally in the form of Bayesian models and ensembles, albeit potentially lacking the calibration guarantee.

ingful uncertainty decomposition, our method offers two key properties:

- **Handles irreducible uncertainty:** It estimates a query's irreducible uncertainty to take appropriate action rather than needlessly escalating. The core routing strategy is straightforward: if the loss is primarily irreducible → abstain (or clarify with the user); if primarily reducible → route to the oracle; otherwise → predict using the weak model.

- **Zero-shot Flexibility:** it can dynamically adapt to different configurations of downstream loss, routing cost, and abstention penalties through simple hyperparameter adjustments, with no need for retraining.

Theoretical properties Our routing framework comes with theoretical guarantees that bound the regret relative to using an *optimal router* that perfectly knows both the weak model's and the oracle's predictions at every point. The strongest form of our guarantee, stated informally, is as follows. Suppose we have a weak model f that we are able to higher-order calibrate with respect to an oracle f^*, which we assume to be the true conditional distribution for our main results, but later generalize. At a high-level, the higher-order calibration guarantee implies that for every weak model prediction v, we have access to the true mixture of $f^*(z)$ ranging over all z such that $f(z) = v$. Then, we can instantiate an uncertainty-aware router that, simultaneously for all task configurations (namely downstream loss, abstention cost, and routing cost), performs at least as well as any task-specific router that is a function of the weak model f. One interpretation of our results is that they provide omniprediction-style guarantees (Gopalan et al., 2022a) for routing. We provide the full technical details in Section 3.

Experiments We run experiments on a variety of real-world and synthetic datasets to obtain a nuanced understanding of the benefits of our method. In Figure 1 we show how our method yields strong routing performance when applied to a synthetic binary classification dataset. However we also observe that in certain scenarios, our method does not yield significant improvements over simpler baselines. To explain this, we lay out different regimes of uncertainty-aware routing, and describe when routing based on reducible uncertainty yields the most benefit. Briefly, we expect to see a benefit when reducible uncertainty is not too correlated with irreducible uncertainty. Experiments are detailed in Section 5.

2. Problem Formulation

2.1. The Routing Framework

We consider a classification setting with instance space $\mathcal{X}$ and discrete label space $\mathcal{Y}$, with the conditional distribution of $y|x$ denoted by $f^*(x) \in \Delta\mathcal{Y}$ (the Bayes-optimal predictor). Here $\Delta\mathcal{Y}$ denotes the space of distributions over $\mathcal{Y}$. We also assume access to a downstream target distribution $\mathcal{D}_{\mathcal{X}}$ that inputs x follow during deployment. This distribution is primarily for calibration purposes, and may differ from the original training distribution for the weak model.

We consider two predictive sources: a low cost, potentially error-prone **weak model** $f : \mathcal{X} \to \Delta\mathcal{Y}$, and a high-cost **oracle** $f^* : \mathcal{X} \to \Delta\mathcal{Y}$ providing the true conditional distribution[2]. For any input x, the router must either **predict** using the weak model, **route** the query to the oracle, or **abstain** from making a prediction altogether (and potentially take fallback actions such as requesting clarification, flagging for later inspection, etc.).

The cost of these actions is determined by a *routing configuration* $T = (L, \alpha, \beta)$, which specifies a bounded loss function $L : \Delta\mathcal{Y} \times \Delta\mathcal{Y} \to [0, B]$, a routing penalty $\alpha \geq 0$, and abstention penalty $\beta \geq 0$. Here $L(p^*, p) := \mathbb{E}_{y\sim p^*}[\ell(y, p)]$ denotes the expected loss of prediction $p \in \Delta\mathcal{Y}$ on a point with ground truth distribution $p^* \in \Delta\mathcal{Y}$. The total cost of each action is a combination of computational overhead and prediction error:

- **Predict:** Using the weak model incurs no computational overhead. The cost is determined solely by the expected prediction loss, $L(f^*(x), f(x))$.

- **Route:** Querying the oracle yields the optimal prediction $f^*(x)$, which incurs loss $L(f^*(x), f^*(x))$. However, this incurs a fixed computational overhead of $\alpha \geq 0$, representing the additional time or resources required to query the stronger model.

- **Abstain:** Declining to predict incurs no computational overhead but results in a fixed penalty of $\beta \geq 0$, representing the cost of failing to provide a response.

The total cost for an action $a \in \{P, R, A\}$ (representing "Predict," "Route," and "Abstain") given an $x \in \mathcal{X}$ and configuration $T = (L, \alpha, \beta)$ is thus:

$$\mathsf{Cost}(x, a; T) := \begin{cases} L(f^*(x), f(x)) & a = P \\ L(f^*(x), f^*(x)) + \alpha & a = R \\ \beta & a = A \end{cases}$$

[2]We note that our framework and results can be extended to a more general pool of oracles whose expected loss can be computed as a function of f^*, a property we term *Bayes-dependency*. See discussion in Appendix D.2 for details.

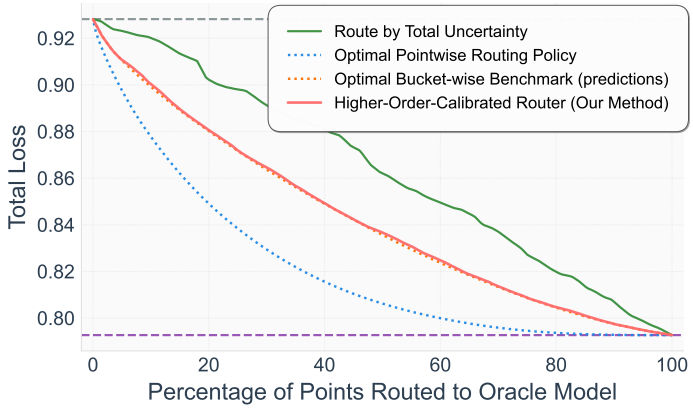


Figure 1. Routing performance of our method on synthetic data and a first-order-calibrated weak model compared to the point-wise optimal routing policy and a router based on total uncertainty. The data generating process is described in Section F.2 and the ground-truth data is visualized in Figure 5a.

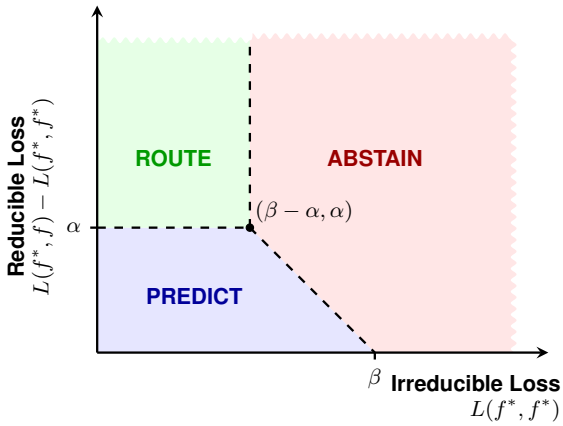


Figure 2. Optimal routing decisions for fixed penalties α and β.

2.2. Optimal Routing and Loss Decomposition

The *pointwise-optimal* routing policy r^* for a configuration $T = (L, \alpha, \beta)$ is the policy that selects the cost-minimizing action for every $x \in \mathcal{X}$:

$$r^*(x, T) := \arg\min_{a \in \{P,R,A\}} \mathsf{Cost}(x, a; T).$$

We now derive the exact form of the optimal router by analyzing the relationship between the weak model's error and the oracle's error. For simplicity, we focus on the case where L is a *proper loss*, meaning that predicting the true conditional distribution minimizes the expected loss, but the general case can be easily reduced to the proper case (see Section H.1 for details).

Given an input x with true conditional distribution $f^*(x)$, the weak model predicts $f(x)$ while the oracle predicts $f^*(x)$. Thus the weak model's loss is $L(f^*(x), f(x))$, while the oracle's loss is $L(f^*(x), f^*(x))$. For a proper loss, the latter is no greater than the former, and so we may decompose the weak model's loss into two key components:

1. The **Irreducible Loss** $L(f^*(x), f^*(x))$: This captures the loss incurred due to inherent noise in the label distribution that even the oracle cannot eliminate.[3]
2. The **Reducible Loss** $L(f^*(x), f(x)) - L(f^*(x), f^*(x))$: This represents the excess loss due to the weak-predictor's sub-optimality, and is exactly the expected difference in loss incurred by trusting the weak model's predictions vs. querying the oracle.

The amount of irreducible loss and reducible loss precisely characterizes which routing action is optimal. Simplifying the cost minimization expression, we can characterize the decisions of the optimal policy in terms of the reducible (RL) and irreducible (IL) loss.

This is illustrated in Figure 2. See Section C for an explanation on how to derive these optimal actions in terms of irreducible and reducible loss.

The optimal policy demonstrates that routing is ineffective for points with high irreducible loss. This explains why naive confidence-based routers often route wastefully; in such cases, the optimal action is typically to abstain or simply trust the weak model.

3. Prediction-Optimal Routing via Higher-Order Calibration

We have seen that the optimal routing decision depends on the *reducible loss* at every point, namely the gap between the weak model's error and the irreducible or aleatoric error. Of course, computing this gap exactly at each point requires access to the true f^*, which we do not have without making an oracle call in the first place. In this section we define a more realistic benchmark for performance: the *prediction-only optimal router*, which represents the best possible router whose decisions are a function of the weak model. We then show that *higher-order calibration* (Ahdritz et al., 2025) provides exactly the necessary uncertainty information to construct a flexible router that competes with the optimal prediction-only policy.

3.1. Prediction-Only Routing

The prediction-only optimal router, $r_f : \mathcal{X} \times \mathcal{T} \to \{P, R, A\}$, is the router that minimizes the routing cost

[3] In fact, $L(f^*(x), f^*(x))$ is precisely the *entropy* $H(f^*(x))$ associated with the proper loss L. For example, when L is the cross-entropy loss, H is the Shannon entropy.

while being restricted to taking the same action on all points given a certain prediction by f (the "level sets" of f). The optimal action for a particular prediction is the one that minimizes the routing cost on average over the level set:

$$r_f(x,T) := \arg\min_{a\in\{P,R,A\}} \mathop{\mathbb{E}}_{x'\sim\mathcal{D}_\mathcal{X}}[\mathsf{Cost}(x',a;T)|f(x')=f(x)].$$

Our goal is to build a flexible router that achieves performance comparable to this relaxed benchmark.

3.2. Why Higher-Order Calibration is Helpful

In contrast to pointwise-optimal routing, determining the prediction-only optimal action only requires good estimates of the reducible loss *on average over each level set*. This requirement is reminiscent of a calibration guarantee. As a reminder, f is said to be calibrated if the average true label matches its predictions over every level set, e.g.

$$p = \mathop{\mathbb{E}}_{x\sim\mathcal{D}_\mathcal{X}, y\sim f^*(x)}[y|f(x)=p].$$

Existing work has shown that calibrated predictors can be used to construct good estimates of the average loss incurred over each level set (Gollakota et al., 2025; Gopalan et al., 2022a). However, estimates of the total loss alone are not enough for our purposes — we need accurate estimates of how the total loss is decomposed into its irreducible and reducible components. To disentangle them, we need the stronger notion of *higher-order calibration*, introduced by Ahdritz et al. (2025).

Higher-order calibration is a property of a *higher-order predictor* $F : \mathcal{X} \to \Delta\Delta\mathcal{Y}$. Unlike standard predictors that output a single label distribution, a higher-order predictor outputs a *distribution over distributions*, quantifying uncertainty about the true $f^*(x)$ rather than just the label y. Higher-order calibration is a form of calibration that guarantees that such higher-order predictions are meaningfully tied to the target distribution:

Definition 3.1 (Higher-Order Calibration, (Ahdritz et al., 2025))**.** A higher-order predictor $F : \mathcal{X} \to \Delta\Delta\mathcal{Y}$ is *ϵ-higher-order calibrated* if for every value $\pi \in \Delta\Delta\mathcal{Y}$ in the range of F, the predicted mixture π is close to the *Bayes-mixture* F^* corresponding to the distribution of $f^*(x)$ for $x \sim \mathcal{D}_\mathcal{X}$, conditioned on $F(x) = \pi$. Formally, for every $\pi \in \text{range}(F)$, ϵ-higher-order-calibration requires that $W_1(\pi, F^*) \le \epsilon$. Here, W_1 denotes the Wasserstein distance with respect to the ℓ_1 norm.

3.3. Main Result: Higher-Order Calibration Enables Optimal Routing

We now show that higher-order calibration is exactly the property needed to construct accurate estimates of the irreducible and reducible loss, on average over level sets. These estimates enable us to make near-optimal routing decisions compared to the prediction-only routing policy.

We make the simplifying assumption that the higher-order predictor, F, shares the same level sets as the weak predictor, f. This means f yields constant predictions across any level set of F. Such a condition arises naturally if F is obtained through post-hoc calibration of the level sets of f. We will explore ways to relax this assumption later in the text. With this assumption in hand, we present our main result:

Theorem 3.1 (Higher-Order Calibration Implies Low Prediction-Only Routing Regret)**.** *Suppose we have an ϵ-higher-order calibrated predictor F whose level sets match those of f. Then we can construct a flexible router r based on F with the following near-optimality property. For any routing configuration $T = (L, \alpha, \beta)$ where L is a B-bounded proper loss and $\alpha, \beta \ge 0$, we have*

$$\mathop{\mathbb{E}}_{x\sim\mathcal{D}_\mathcal{X}}[\mathsf{Cost}(x, r(x,T); T) - \mathsf{Cost}(x, r_f(x,T); T)] \le B\epsilon.$$

The proof of this result can be found in Appendix D.1. Note that we do not require anything about the predictions of the weak model, such as calibration.

4. Practical Higher-Order Predictors

In Section 3, we established higher-order-calibrated predictors as a key tool that can be leveraged to build a flexible routing system that can adapt to different configurations $T = (L, \alpha, \beta)$. In this section, we discuss how to construct such a higher-order calibrated predictor in a practical manner. This requires addressing two engineering challenges: (1) collecting a suitable calibration post-processing dataset, and (2) designing an efficient partition/binning of the input space. Our approach is based on methods for post-hoc higher-order calibration proposed by Ahdritz et al. (2025). We refer the reader to their paper for further discussion of alternative higher-order calibration methods.

4.1. Basic Higher-Order Calibration Approach

To estimate the mixture of conditional expectations $f^*(x)$ within a level set, we cannot rely on standard single-label validation data. We instead require a *k-snapshot calibration set*: a small dataset where each input x is paired with *multiple* independent labels sampled as $y_1, \ldots, y_k \sim f^*(x)$ (e.g., multiple expert annotations). The mean of a k-snapshot, which we denote $\overline{y} := \frac{1}{k}\sum_{i=1}^k y_i \in \Delta\mathcal{Y}$ (where the labels y_i are viewed as one-hot vectors in $\Delta\mathcal{Y}$), serves as an esti-

mate of the true conditional distribution $f^*(x)$. The core of our method lies in using the *empirical mixture* over such $\overline{y}$ as an estimate of the true mixture of $f^*(x)$ over a particular bin.

Specifically, our post-hoc higher-order calibration method uses a "Bin-and-Estimate" procedure:

1. **Partition:** Define a partition $\mathcal{P}$ of the input space $\mathcal{X}$. These are the "bins" (or level sets) on which we will make decisions.

2. **Estimate:** For each bin $P \in \mathcal{P}$, save the empirical distribution (aka normalized histogram) of (prediction, snapshot mean) tuples $(f(x), \overline{y})$ falling within that bin. We denote this empirical distribution S_P. Note that the empirical distribution of $\overline{y}$ is precisely a mixture distribution; what we save as S_P can be thought of as a *tagged* mixture where the mixture components are tagged with an $f(x)$ value.

3. **(Optional Re-Calibration):** Update the values of f to be the centroid of the true labels in its corresponding bin. For $x \in P$,

$$f(x) := \frac{1}{|S_P|} \sum_{(f(x'),\overline{y}) \in S_P} \overline{y}.$$

   Make sure to update the saved predictions in S_P as well.

4. **Route:** For a new query x and for any given routing configuration $T = (L, \alpha, \beta)$, find the bin P containing x. Then use the saved empirical distribution S_P to estimate the expected reducible and irreducible loss, and make the appropriate decision to predict, route, or abstain:

$$\widehat{\mathrm{IL}} := \frac{1}{|S_P|} \sum_{(f(x'),\overline{y}) \in S_P} L(\overline{y}, \overline{y})$$

$$\widehat{\mathrm{RL}} := \frac{1}{|S_P|} \sum_{(f(x'),\overline{y}) \in S_P} L(\overline{y}, f(x')) - \widehat{IL}.$$

$$r(x, T) = \arg\min_{a \in \{P,R,A\}} \begin{cases} \widehat{\mathrm{IL}} + \widehat{\mathrm{RL}} & a = P \\ \widehat{\mathrm{IL}} + \alpha & a = R \\ \beta & a = A \end{cases}$$

Notice that the statistics saved (namely S_P for each bin P) are completely independent of the routing configuration, which enters only in step 4. This is the source of the flexibility of our approach: for any given configuration, step 4 can be carried out cheaply using the saved statistics.

Remark 4.1. While the empirical mean $\frac{1}{k}\sum_{i=1}^{k} y_i$ concentrates around $f^*(x)$ for large k, using it as a proxy when k is small yields biased loss estimates. In such cases, we recommend the unbiased estimation techniques from (Ahdritz et al., 2025) (e.g. Theorem 4.3) to accurately decompose the irreducible and reducible loss. For our experiments we work with the empirical mean for simplicity.

4.2. Selecting a Partition

The success of this framework hinges on the choice of partition $\mathcal{P}$, which governs the tradeoff between granularity and estimation stability. While fine-grained partitions (e.g. exact level sets) allow for precise routing decisions, they often suffer from data sparsity, meaning that there may not be sufficient data to have good estimates of the loss decomposition in each partition (especially in multiclass prediction, since the number of level sets scales exponentially with the number of classes). Coarser partitions ensure each bin has enough samples for a stable estimate, but they force the router to apply the same policy to broad regions of the input space.

Even restricting to partitions that are coarse enough to have enough samples for each cell, many ways to divide the space remain. At a high level, a good partition should try to *minimize the variance of the reducible and irreducible loss* within each cell. We offer a formalization of this intuition in Appendix E along with a discussion of recommended strategies for selecting buckets.

5. Experiments

In this section, we implement our routing framework following the method described in Section 4, and study its performance on a number of synthetic and real-world datasets.

Direct comparison to existing baselines is nuanced, as our method targets a broader scope with three primary advantages. First, unlike confidence-based methods that consider only total uncertainty, our framework routes effectively under irreducible uncertainty, avoiding unnecessary oracle queries. Second, it offers flexibility, providing optimal routing for any proper loss without retraining, whereas supervised methods are fixed during training. Finally, it supports a richer action space—predicting, routing, or abstaining—while existing frameworks are generally limited to two options and are difficult to extend.

To demonstrate these different benefits of our routers, we divide our experiments into three parts:

1. **Routing with Irreducible Uncertainty**: In this section, we demonstrate our router's ability to distinguish between hard points with high reducible loss that should be routed, vs ambiguous points with high

irreducible loss that will not benefit from routing, a distinction that confidence-based methods fail to make. For this section, we focus on the simplified routing setting where the available actions are only to route or predict.

2. **Loss Flexibility**: In this section, we study the brittleness of supervised methods in settings with dynamically changing losses. We show that our method can adapt without the need for retraining to a variety of loss functions, while supervised methods degrade on losses that differ from the fixed configuration used during training.

3. **Three-way Routing Decisions**: In this section, we show that our method can gracefully expand beyond 2-way decisions (route vs. predict) to accommodate a third option: abstention.

5.1. Experiment Setup

5.1.1. Datasets and Model Architectures

We evaluate our method on a synthetic binary prediction task and three real-world datasets characterized by inherent human label ambiguity: CIFAR-10H, SNLI, and ChaosNLI. For each dataset, we train a "weak predictor" f whose prediction we will use when deciding whether to route to an oracle or not. See Section F.1 for detailed information on each dataset and its associated weak predictor.

Oracle Predictor (f^*) For each dataset and tested loss, we compute the loss of the oracle predictor using the label distribution defined per dataset. For example, if we are testing CIFAR-10H with the cross entropy loss, the oracle loss would be the entropy of the label distribution of the 47 human annotations on the particular input. We have intentionally chosen datasets with varying amounts of intrinsic aleatoric uncertainty. Therefore, the oracle loss will usually be larger than zero, implying some disagreement exists between human labelers.

Routing Framework We implement our routing framework using the post-hoc "Bin-and-Estimate" procedure detailed in Section 4. To ensure sample efficiency, we employ a top-class binning approach, defining partitions based on the predicted class and its corresponding confidence score. For each class, we divide these probabilities into 10 quantile-based buckets, ensuring an equal distribution of calibration points across bins.

5.1.2. Baselines and Evaluation

Two-way Routing Curves While our method naturally handles three-way decisions (predict, route, or abstain), existing frameworks such as Hendrickx et al. (2024); El-Yaniv et al. (2010) are typically restricted to binary actions (predict/route or predict/abstain). To ensure a fair comparison, our first two experiments focus on these two-way settings using cost-agnostic routing curves, which plot the percentage of points routed (x-axis) against total loss (y-axis).

This evaluation is deliberately favorable to baselines: by sweeping across all possible thresholds, the curve effectively grants every method its optimal post-hoc operating point. While our framework is designed to explicitly optimize for specific penalties α and β, the curve treats these costs as unknown, selecting the loss-minimizing threshold for every point on the plot. Any strategy that orders points by priority can generate such a curve; superior strategies yield sharply declining curves, while random selection results in a linear decrease.

Lastly, in our third experiment, we demonstrate that our method seamlessly adapts to three-way decisions and variable costs (scenarios where simple thresholding often fails).

Routing Strategies and Baselines We compare our method's routing curves against several established strategies and optimal benchmarks:

- **Route by Total Uncertainty**: Points are ranked by their predictive entropy $H(f(x)) = L(f(x), f(x))$, where H corresponds to the target loss. Points with high entropy are prioritized for routing.
- **Supervised Router**: To evaluate our framework against a direct supervised alternative, we designed a regression-based router specifically for these tasks. This model — an XGBoost Regressor — is trained on the calibration dataset to predict the amount of reducible loss at each point, using the prediction $f(x)$ and input features x to estimate the gap $\ell(y, f(x)) - \ell(y, f^*(x))$. While this represents a natural supervised approach to our routing objective, it is inherently limited in two ways. First, while it performs well for binary decisions (e.g., route vs. predict)—often matching or even outperforming our method on synthetic data—it cannot be directly extended to multi-way routing (e.g., adding an "abstain" option). Second, the regressor is tied to the specific loss used during training, whereas our method remains flexible across a variety of losses without retraining (see Benefit 2).
- **Point-wise Optimal Router**: An optimal baseline that orders points based on the true reducible loss, $L(f^*(x), f(x)) - L(f^*(x), f^*(x))$.
- **Bucket-Optimal Router** (occasionally referred to as **Prediction-Optimal Router** for simplicity): A relaxed baseline representing the best possible routing decisions that remain constant over the buckets used in our framework. This router measures the average reducible loss within each bucket on the test set and orders points accordingly.

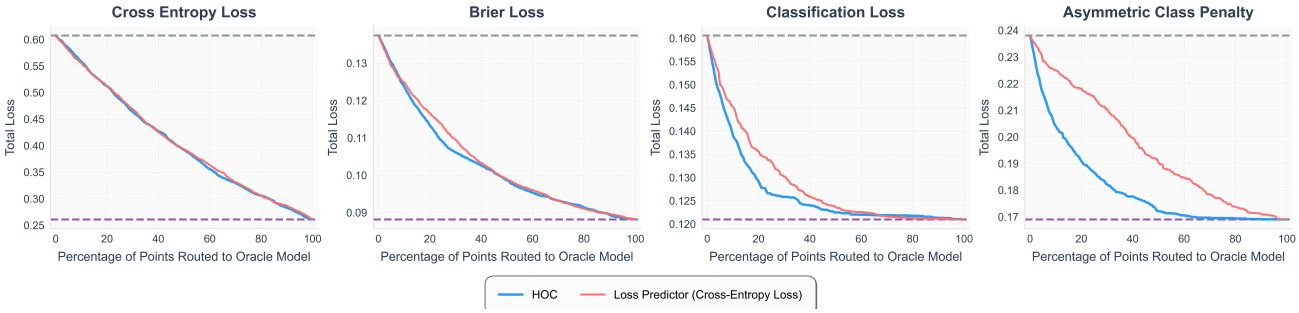


Figure 3. Routing performances for different target loss functions for the SNLI dataset. The loss predictor baseline is a static loss predictor trained to predict the amount of reducible cross-entropy loss. While our method (HOC) performs roughly comparably to the loss predictor when routing based on cross-entropy loss, we see the performance of the loss predictor degrade when deployed on new losses further from the cross-entropy loss, whereas our method maintains good performance.

5.2. Benefit 1: Routing in the Presence of Irreducible Uncertainty

For simplicity, we first consider a binary routing setting: the system must either predict using the base model or route to the oracle, which in this case predicts the true ground-truth conditional probability. In this simplified setting, the optimal router routes any point with reducible loss that is above a particular threshold. As discussed in Section 2.2, routing strategies that fail to decompose the amount of reducible vs irreducible loss may fail in settings where the two quantities are de-coupled.

We demonstrate this advantage in Figure 1. In this synthetic environment, routing based on total uncertainty of even a well-calibrated predictor is significantly outperformed by our higher-order method. By explicitly estimating and routing based on the reducible loss, our framework avoids "wasting" the routing budget on inherently ambiguous points that the oracle cannot improve upon.

When Decomposition Matters The practical benefit of separating irreducible and reducible loss is intrinsically tied to the joint distribution of these two quantities. We note that both the structure of the data and weak predictions as well as the bucket choice for binning-based methods such as ours can greatly affect to what degree decomposition is helpful for routing. See Section F.3.1 for further discussion of these effects.

Observations on Real-World Datasets We extended this analysis to several real-world datasets characterized by human label ambiguity: CIFAR-10H, SNLI, and ChaosNLI. Interestingly, we observed that in these datasets, total and irreducible loss remain highly correlated across the level sets of standard models. Consequently, decomposition did not yield the same dramatic performance gaps seen in our synthetic experiments.

This correlation persists in spite of non-trivial variation in the amount of irreducible loss in these datasets, suggesting that it is likely a product of either the training process or our bucketing approach rather than the data itself. We note that this finding aligns with prior observations by (Mucsányi et al., 2024), who noted a high empirical correlation between aleatoric and epistemic uncertainties in trained models.

We provide examples of this behavior in Figures 9a, 10a and 11a on the CIFAR-10H, SNLI, and ChaosNLI datasets, respectively. When the base model is well-calibrated, most methods converge toward the bucket-optimal routing curve. This suggests that in some settings routing by total uncertainty may be a sufficient proxy for estimates of reducible loss. However, our synthetic demonstrations highlight that determining whether decomposition is important should be a key first step when implementing a router.

Remark 5.1. We observed that many of the models we trained were significantly miscalibrated when used out of the box, meaning that our method performed significantly better than the total uncertainty baseline. This highlights that our method can continue to perform well even in settings where the model is miscalibrated. However, the fact that we observe these benefits collapsing after calibrating the model suggests that most of the observed improvement comes from miscalibration, rather than a failure to decompose irreducible and reducible loss.

5.3. Benefit 2: Loss Flexibility

A key benefit of our framework is its flexibility. In particular, it can quickly adapt to new routing penalties, loss functions, and even oracles. In this section, we focus on its flexibility with respect to loss functions. We compare to a natural baseline – a supervised "loss predictor," which takes as input a prediction and features about an input, and outputs a prediction of the amount of reducible loss.

In a static setting where the loss is fixed, this supervised approach tends to perform very well, even out-performing our partition-based method on some datasets. However, in

dynamic settings where the loss we care about may change, the downside of this approach is that we must retrain a new loss predictor for each new loss. Figure 3 depicts the potential loss in performance resulting from using a single statically trained loss predictor to make routing decisions on new losses (detailed in Section H.1) on the SNLI dataset. In contrast, our method preserves performance guarantees across a variety of different loss functions. We present similar plots for all datasets in Section H.

5.4. Benefit 3: Effective Three-Way Decisions

Thus far, we have focused on two-way decisions between prediction and routing. However, a key benefit of our method is that it can gracefully adapt to multi-way decisions beyond simply choosing whether to route or predict. In particular, in this section we demonstrate its ability to decide not only between prediction and routing, but also whether to abstain when the irreducible loss is too high, where the optimal decision for this setting is visualized in Figure 2.

Figure 4 depicts the cost-penalized loss of our method on the SNLI dataset, with similar plots for other datasets included in Section I. In this plot, the routing cost (α) is kept fixed and the abstention cost (β) is swept from 0.1 to 0.8. We compare the full three-way power of our method against restricted two-way strategies (predict vs. route, and predict vs. abstain) using our calibrated estimates of the irreducible and total loss, respectively. As shown, the ability of our method to extend to three-way decisions allows it to achieve lower loss than a simpler framework that allows for only two options, effectively tracing the lower envelope of the two-way decision curves.

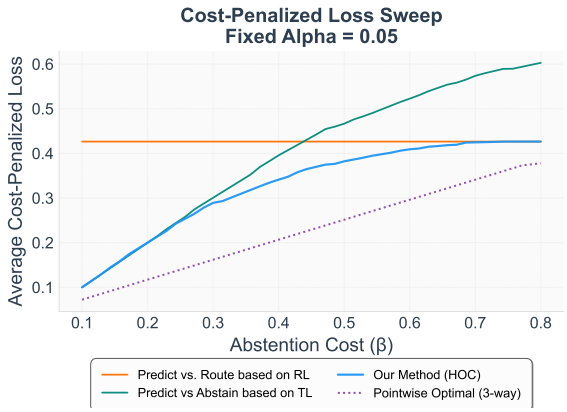


Figure 4. Cost-penalized performance of our method on the three-way decision task (Predict vs. Route vs. Abstain) on the SNLI dataset, for a fixed routing penalty of $\alpha = 0.05$ while sweeping the abstention penalty β. Our method gracefully adapts to the three-way decision setting, out-performing the two-way decision baselines of predict/route (orange) and predict/abstain (green).

6. Conclusions

This work demonstrates the need for theoretically rigorous routing frameworks capable of adapting to unanticipated downstream desiderata such as new loss functions, costs, or routing oracles. We highlighted the importance of uncertainty decomposition for optimal routing, and proposed a practical and provably effective routing mechanism based on higher-order calibrated predictors (Ahdritz et al., 2025), validated by empirical experiments. We view developing more efficient methods for learning high-quality higher-order-calibrated predictors as a key direction of future work that could further enhance the scalability of our framework.

Acknowledgements

CP is supported by the Simons Foundation Collaboration on the Theory of Algorithmic Fairness and the Apple Scholars in AI/ML PhD fellowship.

Impact Statement

This paper presents work whose goal is to advance the field of Machine Learning. There are many potential societal consequences of our work, none which we feel must be specifically highlighted here.

References

Ahdritz, G., Gollakota, A., Gopalan, P., Peale, C., and Wieder, U. Provable uncertainty decomposition via higher-order calibration. In *The Thirteenth International Conference on Learning Representations*, 2025.

Alur, R., Laine, L., Li, D. K., Shung, D., Raghavan, M., and Shah, D. Integrating expert judgment and algorithmic decision making: An indistinguishability framework. *arXiv preprint arXiv:2410.08783*, 2024a.

Alur, R., Raghavan, M., and Shah, D. Human expertise in algorithmic prediction. *Advances in Neural Information Processing Systems*, 37:138088–138129, 2024b.

Awasthi, P., Mao, A., Mohri, M., and Zhong, Y. H-consistency bounds for surrogate loss minimizers. In *International Conference on Machine Learning*, pp. 1117–1174. PMLR, 2022.

Bowman, S., Angeli, G., Potts, C., and Manning, C. D. A large annotated corpus for learning natural language inference. In *Proceedings of the 2015 conference on empirical methods in natural language processing*, pp. 632–642, 2015.

Casacuberta, S. and Kanade, V. Selective omniprediction and fair abstention. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025.

Chow, C.-K. An optimum character recognition system using decision functions. *IRE Transactions on Electronic Computers*, (4):247–254, 1957.

Cortes, C., DeSalvo, G., and Mohri, M. Learning with rejection. In *International conference on algorithmic learning theory*, pp. 67–82. Springer, 2016.

Dekoninck, J., Baader, M., and Vechev, M. A unified approach to routing and cascading for llms. *arXiv preprint arXiv:2410.10347*, 2024.

Devlin, J., Chang, M., Lee, K., and Toutanova, K. BERT: pre-training of deep bidirectional transformers for language understanding. *CoRR*, abs/1810.04805, 2018. URL http://arxiv.org/abs/1810.04805.

Ding, D., Mallick, A., Wang, C., Sim, R., Mukherjee, S., Rühle, V., Lakshmanan, L. V., and Awadallah, A. H. Hybrid llm: Cost-efficient and quality-aware query routing. In *The Twelfth International Conference on Learning Representations*, 2024.

Ding, D., Mallick, A., Zhang, S., Wang, C., Madrigal, D., Garcia, M. D. C. H., Xia, M., Lakshmanan, L. V., Wu, Q., and Rühle, V. Best-route: Adaptive llm routing with test-time optimal compute. *arXiv preprint arXiv:2506.22716*, 2025.

Donahue, K., Chouldechova, A., and Kenthapadi, K. Human-algorithm collaboration: Achieving complementarity and avoiding unfairness. In *Proceedings of the 2022 ACM Conference on Fairness, Accountability, and Transparency*, pp. 1639–1656, 2022.

El-Yaniv, R. et al. On the foundations of noise-free selective classification. *Journal of Machine Learning Research*, 11 (5), 2010.

Enomoro, S. and Eda, T. Learning to cascade: Confidence calibration for improving the accuracy and computational cost of cascade inference systems. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pp. 7331–7339, 2021.

Geifman, Y. and El-Yaniv, R. Selective classification for deep neural networks. *Advances in neural information processing systems*, 30, 2017.

Gollakota, A., Gopalan, P., Karan, A., Peale, C., and Wieder, U. When does a predictor know its own loss? In *6th Symposium on Foundations of Responsible Computing (FORC 2025)*, pp. 22–1. Schloss Dagstuhl–Leibniz-Zentrum für Informatik, 2025.

Gopalan, P., Hu, L., Kim, M. P., Reingold, O., and Wieder, U. Loss minimization through the lens of outcome indistinguishability. *arXiv preprint arXiv:2210.08649*, 2022a.

Gopalan, P., Kalai, A. T., Reingold, O., Sharan, V., and Wieder, U. Omnipredictors. In *13th Innovations in Theoretical Computer Science Conference (ITCS 2022)*, pp. 79–1. Schloss Dagstuhl–Leibniz-Zentrum für Informatik, 2022b.

He, K., Zhang, X., Ren, S., and Sun, J. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016.

Hendrickx, K., Perini, L., Van der Plas, D., Meert, W., and Davis, J. Machine learning with a reject option: A survey. *Machine Learning*, 113(5):3073–3110, 2024.

Jitkrittum, W., Narasimhan, H., Rawat, A. S., Juneja, J., Wang, C., Wang, Z., Go, A., Lee, C.-Y., Shenoy, P., Panigrahy, R., et al. Universal model routing for efficient llm inference. *arXiv preprint arXiv:2502.08773*, 2025.

Johnson, D. D., Tarlow, D., Duvenaud, D., and Maddison, C. J. Experts don't cheat: Learning what you don't know by predicting pairs. In Salakhutdinov, R., Kolter, Z., Heller, K. A., Weller, A., Oliver, N., Scarlett, J., and Berkenkamp, F. (eds.), *Forty-first International Conference on Machine Learning, ICML 2024, Vienna, Austria, July 21-27, 2024*, volume 235 of *Proceedings of Machine Learning Research*, pp. 22406–22464. PMLR / OpenReview.net, 2024. URL https://proceedings.mlr.press/v235/johnson24a.html.

Mao, A., Mohri, C., Mohri, M., and Zhong, Y. Two-stage learning to defer with multiple experts. *Advances in neural information processing systems*, 36:3578–3606, 2023.

Mao, A., Mohri, M., and Zhong, Y. Theoretically grounded loss functions and algorithms for score-based multi-class abstention. In *International Conference on Artificial Intelligence and Statistics*, pp. 4753–4761. PMLR, 2024.

Mohri, C., Andor, D., Choi, E., and Collins, M. Learning to reject with a fixed predictor: Application to decontextualization. *arXiv preprint arXiv:2301.09044*, 2023.

Mucsányi, B., Kirchhof, M., and Oh, S. J. Benchmarking uncertainty disentanglement: Specialized uncertainties for specialized tasks. *Advances in neural information processing systems*, 37:50972–51038, 2024.

Nie, Y., Williams, A., Dinan, E., Bansal, M., Weston, J., and Kiela, D. Adversarial NLI: A new benchmark for natural language understanding. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics, 2020a.

Nie, Y., Zhou, X., and Bansal, M. What can we learn from collective human opinions on natural language inference data? In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Association for Computational Linguistics, 2020b.

Okoroafor, P., Kleinberg, R., and Kim, M. P. Near-optimal algorithms for omniprediction. In *2025 IEEE 66th Annual Symposium on Foundations of Computer Science (FOCS)*, pp. 1595–1609, 2025. doi: 10.1109/FOCS63196.2025.00084.

Ong, I., Almahairi, A., Wu, V., Chiang, W.-L., Wu, T., Gonzalez, J. E., Kadous, M. W., and Stoica, I. Routellm: Learning to route llms from preference data. In *The Thirteenth International Conference on Learning Representations*, 2025.

Peterson, J. C., Battleday, R. M., Griffiths, T. L., and Russakovsky, O. Human uncertainty makes classification more robust. In *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 9617–9626, 2019.

Raghu, M., Blumer, K., Corrado, G., Kleinberg, J., Obermeyer, Z., and Mullainathan, S. The algorithmic automation problem: Prediction, triage, and human effort. *arXiv preprint arXiv:1903.12220*, 2019.

Somerstep, S., Polo, F. M., de Oliveira, A. F. M., Mangal, P., Silva, M., Bhardwaj, O., Yurochkin, M., and Maity, S. Carrot: A cost aware rate optimal router. *arXiv preprint arXiv:2502.03261*, 2025.

Strong, J., Sun, E., Rogers, H., Higham, H., and Alison, J. Learning to defer: A survey. 2025.

Varangot-Reille, C., Bouvard, C., Gourru, A., Ciancone, M., Schaeffer, M., and Jacquenet, F. Doing more with less–implementing routing strategies in large language model-based systems: An extended survey. *arXiv preprint arXiv:2502.00409*, 2025.

Williams, A., Nangia, N., and Bowman, S. A broad-coverage challenge corpus for sentence understanding through inference. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pp. 1112–1122. Association for Computational Linguistics, 2018. URL http://aclweb.org/anthology/N18-1101.

A. Additional Related Work

Model routing and related problems such as orchestration, learning to defer, and selective prediction have been the subject of a vast body of work. Here we do not attempt a complete survey but instead aim to highlight the general principles of these lines of work and how they relate to ours. We note at the outset that our method is not intended to be fully general-purpose but rather assumes: (a) a binary or multiclass classification setting with access to a calibration dataset consisting of multiple labels per input, (b) an oracle that is strictly stronger than (not complementary to) the weak model. A significant difference from all prior work is that to our knowledge, ours is the first method that performs both abstention and routing in a unified way, and with strong distribution-free theoretical guarantees.

Selective prediction. The idea of augmenting machine learning models with an option to abstain (or reject) dates back at least to (Chow, 1957); see (Hendrickx et al., 2024) for a modern survey. The cost of abstaining is usually modeled as a fixed cost (as we do), with the goal of a selective predictor being to minimize this modified loss function. Since it is typically nonconvex, many works have studied minimizing suitable surrogates instead (see e.g. (Cortes et al., 2016; Mao et al., 2024)). The decision to abstain is made either based on the model's own confidence or a separately supervised "rejector" model. Recent literature in this area has proposed evaluating selective predictors in terms of their risk-coverage curves, and various works have studied optimal strategies and inherent tradeoffs under certain idealized settings (El-Yaniv et al., 2010; Geifman & El-Yaniv, 2017). In the abstention-only setting our router specializes to a confidence-based policy that is justified by a higher-order form of calibration.

Learning to defer. Here the goal is precisely to learn to defer to an oracle on inputs where there is a benefit to doing so; see (Strong et al., 2025) for a survey. Routing to the oracle is modeled as incurring an instance-dependent cost which may be a function of oracle's loss, but may in general be arbitrary. For this reason learning to defer is often viewed as a generalization of selective prediction where the abstain option carries an adaptive or instance-dependent penalty rather than a fixed one. The focus of many papers in this area is on tractable approaches for minimizing the resulting modified cost function, or surrogates thereof, usually over a fixed model class (see e.g. (Awasthi et al., 2022; Mohri et al., 2023; Mao et al., 2023)). In our setting we effectively consider the special case where the oracle has a known form, and the cost incurred upon routing is the oracle's loss plus a fixed routing penalty.

Model cascades. In this approach we run a suite of models in a cascade ordered by increasing cost, passing the output each time to an auxiliary verifier or quality estimation model, and stop when the output meets a desired threshold (see e.g. (Enomoro & Eda, 2021)). In this sense cascading may be considered a sequential learning to defer problem.

Orchestration. Unlike our setting, orchestration considers a suite of models with complementary strengths, and the goal is to learn to route to the most suitable model for a given query. As there is no relationship between the models, this approach largely rests on using supervision to train a router. The Mixture of Experts (MoE) architecture can be seen as a version of this paradigm where the models (or experts) and router (or gating function) are trained in tandem. In some variants, routing is also done based on clustering queries in embedding space. See e.g. (Dekoninck et al., 2024; Jitkrittum et al., 2025) and references therein.

Loss Prediction, Omniprediction The ability of our router to dynamically adapt to new losses and routing penalty draws on work on omnipredictors (Gopalan et al., 2022b), which presents a way to learn a single predictor that can be post-processed to minimize any loss from a rich class of losses. Follow-up work has more explicitly made the connection between the ability to accurately estimate a model's loss and various calibration conditions (Gopalan et al., 2022a; Gollakota et al., 2025; Okoroafor et al., 2025). Omniprediction has also been extended to the selective abstention setting by (Casacuberta & Kanade, 2025). Our work can be viewed as a higher-order extension of these ideas, allowing us not only to estimate the overall loss, but decompose it into irreducible and reducible components in a way that enables more principled routing decisions.

Complementarity, Human-AI Collaboration Our work is also related to a line of theoretical results from Human-AI collaboration. Works such as Raghu et al. (2019); Donahue et al. (2022); Alur et al. (2024b;a) study ways of *combining* human and algorithmic predictions. Importantly, this requires understanding when and where predictions from different sources are *complementary* and can be combined to improve overall decision quality. In contrast, our work considers routing to an oracle model which we imagine to be strictly more powerful than the cheaper model everywhere.

Routing in LLMs There are a number of papers related to routing in the LLM literature (Ding et al., 2024; 2025; Ong et al., 2025). These works usually consider the problem of routing a user query to one of a number of different LLMs which may differ on quality, cost per token, or domain expertise. The overall goal of LLM routing is to maximize performance

subject to a given budget either per query or aggregated over the entire dataset. Jitkrittum et al. (2025) propose a router for an arbitrary set of LLMs which embeds both LLMs and prompts / queries in a particular feature space to make routing decisions. Their approach treats the possible LLMs as a *dynamic pool*, which then allows the router to still function without re-training when certain LLMs are deprecated, or new LLMs are added to the pool. Somerstep et al. (2025) point out that any optimal LLM routing approach should take into account both the predictive power of each LLM, as well as their cost per token. They use this insight to develop algorithms and lower bounds for LLM routing based on predicting both cost and accuracy. For a comprehensive survey on LLM routing, we refer to (Varangot-Reille et al., 2025). More generally, in contrast to our approach, most LLM routing papers rarely explicitly consider epistemic and aleatoric uncertainty decompositions, and simply optimize the pareto-frontier of certain performance-cost curves.

B. Useful Results about Losses

In this section, we prove a few useful results about bounded proper losses that will be useful in the proofs used throughout the paper. Given a loss $\ell : \mathcal{Y} \times \Delta\mathcal{Y} \to [0, B]$, we denote its expectation over a conditional distribution as $L(p^*, p) = \mathbb{E}_{y\sim p^*}[\ell(y, p)]$.

We say that a function $g : \Delta\mathcal{Y} \to \mathbb{R}$ satisfies L-Lipschitz continuity if for every $p_1, p_2 \in \Delta\mathcal{Y}$,

$$g(p_1) - g(p_2) \leq L\|p_1 - p_2\|_1.$$

The first lemma shows that any bounded proper loss satisfies Lipschitz continuity in its first argument.

Lemma B.1. *Let L be a proper loss with range $[0, B]$. For any $p_1, p_2, q \in \Delta\mathcal{Y}$, L satisfies*

$$|L(p_1, q) - L(p_2, q)| \leq \frac{B}{2}\|p_1 - p_2\|_1.$$

Proof. Let $H'(q)$ denote the vector such that $H'(q)_y := \ell(y, q)$ for each $y \in \mathcal{Y}$. We use this vector to equivalently re-write our loss expression as an inner-product over $\mathcal{Y}$:

$$|L(p_1, q) - L(p_2, q)| = |\langle p_1 - p_2, H'(q)\rangle|.$$

We note that because p_1 and p_2 are both probability distributions over $\mathcal{Y}$, we have that $\langle p_1 - p_2, \mathbf{1}\rangle = 0$, and so for any $c \in \mathbb{R}$, we have

$$\begin{aligned}|\langle p_1 - p_2, H'(q)\rangle| &= |\langle p_1 - p_2, H'(q) - c\mathbf{1}\rangle| \\ &\leq \|p_1 - p_2\|_1 \|H'(q) - c\mathbf{1}\|_\infty\end{aligned}$$

We select the c that minimizes $\|H'(q) - c\mathbf{1}\|_\infty$, $c = \frac{1}{2}\left(\max_{y\in\mathcal{Y}} H'(q)_y + \min_{y'\in\mathcal{Y}} H'(q)_{y'}\right)$. This choice of c gives an infinity norm value of

$$\begin{aligned}\|H'(q) - c\mathbf{1}\|_\infty &= \frac{1}{2}\left(\max_{y\in\mathcal{Y}} H'(q)_y - \min_{y'\in\mathcal{Y}} H'(q)_{y'}\right) \\ &= \frac{1}{2}\left(\max_{y\in\mathcal{Y}} \ell(y, q) - \min_{y'\in\mathcal{Y}} \ell(y', q)\right) && \text{(Definition of } H'(q)\text{)} \\ &\leq \frac{B}{2} && \text{(Range of } \ell \text{ is } [0, B].\text{)}\end{aligned}$$

Substituting this back into our upper bound, we conclude that

$$|L(p_1, q) - L(p_2, q)| \leq \frac{B}{2}\|p_1 - p_2\|_1.$$

□

We use Lemma B.1 to prove that two loss-related quantities satisfy $B/2$-Lipschitz continuity.

Lemma B.2. *For any fixed $p \in \Delta\mathcal{Y}$ and bounded proper loss $L : \Delta\mathcal{Y} \times \Delta\mathcal{Y} \to [0, B]$, the functions $g_{TL}(p^*) = L(p^*, p)$ and $g_{IR}(p^*) = L(p^*, p^*)$ satisfy $B/2$-Lipschitz continuity.*

Proof. The $B/2$-Lipschitz continuity of g_{TL} follows immediately from Lemma B.1. For g_{IR}, we note that because L is a proper loss, we have that for any p_1 and p_2,

$$\begin{aligned} g_{IR}(p_1) - g_{IR}(p_2) &= L(p_1, p_1) - L(p_2, p_2) \\ &\leq L(p_1, p_2) - L(p_2, p_2) && (L \text{ is proper}) \\ &\leq \frac{B}{2}\|p_1 - p_2\|_1 && (\text{Lemma B.1}) \end{aligned}$$

and thus g_{IR} also satisfies $B/2$-Lipschitz continuity. □

C. Connecting Optimal Routing Decisions to Reducible Loss

In this section, we provide more explanation on how the optimal routing decisions are determined from the amount of reducible and irreducible loss, as illustrated in Figure 2.

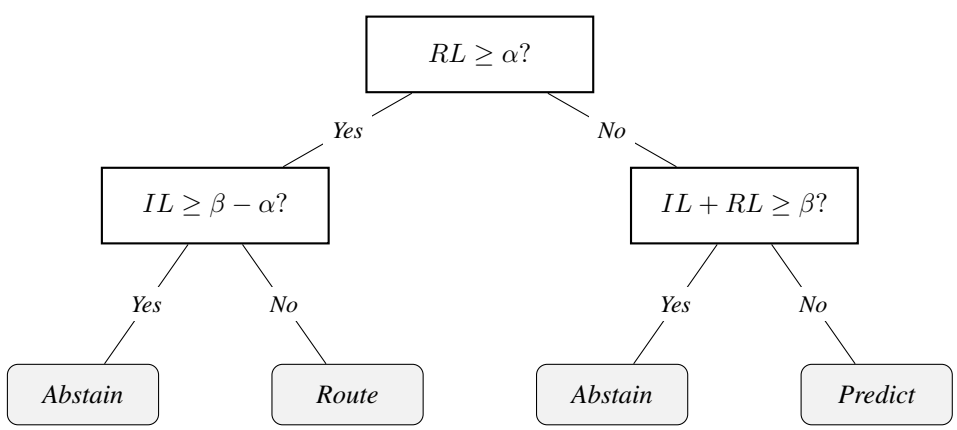
Proposition C.1. *Given a point x with irreducible loss and reducible loss denoted*

$$\text{IL} := L(f^*(x), f^*(x)), \quad \text{RL} := L(f^*(x), f(x)) - L(f^*(x), f^*(x)),$$

respectively, the point-wise optimal routing decision for any $\alpha, \beta \geq 0$,

$$r^*(x, (L, \alpha, \beta)) = \arg\min_{a \in \{P,R,A\}} \mathsf{Cost}(x, a; (L, \alpha, \beta))$$

is equivalent to the following decision framework in terms of IL *and* RL*:*



Proof. Based on our definition of the routing cost, routing has a lower cost than prediction precisely when

$$\begin{aligned} \mathsf{Cost}(x, R; T) &\leq \mathsf{Cost}(x, P; T) \\ L(f^*(x), f^*(x)) + \alpha &\leq L(f^*(x), f(x)) \\ \alpha &\leq L(f^*(x), f(x)) - L(f^*(x), f^*(x)) \\ \alpha &\leq \text{RL}. \end{aligned}$$

Thus, this reconstructs the first layer of the tree – routing is preferred over prediction by the optimal router if and only if $\text{RL} \geq \alpha$.

Assuming $\mathrm{RL} \geq \alpha$, we conclude that the optimal router will either abstain or route. The property that abstention has a lower cost than routing can be simplified as follows:

$$\begin{aligned} \mathsf{Cost}(x, A; T) &\leq \mathsf{Cost}(x, R; T) \\ \beta &\leq L(f^*(x), f^*(x)) + \alpha \\ \beta - \alpha &\leq \mathrm{IL} \end{aligned}$$

Thus, we have recovered the lower left side of the tree – assuming $\mathrm{RL} \geq \alpha$, the optimal router abstains whenever $\mathrm{IL} \geq \beta - \alpha$, and routes otherwise.

We finally consider the other case: $\mathrm{RL} \leq \alpha$. We know that in this case, the optimal routing policy will either predict or abstain. Abstention has a lower cost compared to prediction precisely when

$$\begin{aligned} \mathsf{Cost}(x, A; T) &\leq \mathsf{Cost}(x, P; T) \\ \beta &\leq L(f^*(x), f(x)) \\ \beta &\leq \mathrm{RL} + \mathrm{IL}. \end{aligned}$$

Thus, we have recovered the lower right side of the tree, and shown that the decisions are exactly the pointwise optimal routing decisions. □

D. Proofs and Additional Discussion from Section 3

D.1. Proof of Theorem 3.1

In this section, we prove Theorem 3.1, which we restate here for readability:

Theorem 3.1 (Higher-Order Calibration Implies Low Prediction-Only Routing Regret)**.** *Suppose we have an ϵ-higher-order calibrated predictor F whose level sets match those of f. Then we can construct a flexible router r based on F with the following near-optimality property. For any routing configuration $T = (L, \alpha, \beta)$ where L is a B-bounded proper loss and $\alpha, \beta \geq 0$, we have*

$$\mathop{\mathbb{E}}_{x \sim \mathcal{D}_{\mathcal{X}}} [\mathsf{Cost}(x, r(x, T); T) - \mathsf{Cost}(x, r_f(x, T); T)] \leq B\epsilon.$$

The proof of the theorem relies on Lemma D.2, which captures why higher-order calibration is so useful when constructing a routing framework. Intuitively, the lemma guarantees that a higher-order-calibrated prediction can be used to obtain good estimates of the routing cost for each potential routing decision. This means that choosing the action that minimizes our estimated cost will also be a good minimizer of the true routing cost.

For ease of exposition, it will be useful to define a generalized routing cost that takes in precise values of $f^*(x) = p^*$ and $f(x) = p$, rather than just x. We define the generalized cost as follows:

$$\mathsf{Cost}(p^*, p, a; T) := \begin{cases} L(p^*, p) & a = P \\ L(p^*, p^*) + \alpha & a = R \\ \beta & a = A \end{cases}$$

Under this new notation, $\mathsf{Cost}(x, a; T)$ can be written equivalently as $\mathsf{Cost}(f^*(x), f(x), a; T)$.

Given a higher-order prediction F, its predictions can be used to estimate the routing cost at each point, where we can interpret the estimates as the routing loss that would be incurred if the true value of $f^*(x)$ was actually distributed according to the higher-order prediction:

Definition D.1 (Simulated Routing Cost)**.** Given a higher-order predictor F, we define the *simulated routing cost* as

$$\widehat{\mathsf{Cost}}_F(x, a; T) := \mathop{\mathbb{E}}_{p^* \sim F(x)} [\mathsf{Cost}(p^*, f(x), a; T)].$$

The following lemma establishes that the simulated routing loss is a good estimate for the true routing cost for any action, on average over each partition.

Lemma D.2 (Routing Loss Prediction from Higher-Order Calibration)**.** *Let F be a ϵ-higher-order-calibrated predictor whose level sets match those of f and let $T = (L, \alpha, \beta)$ be a routing configuration where L is a B-bounded proper loss $L : \Delta\mathcal{Y} \times \Delta\mathcal{Y} \to [0, B]$, and $\alpha, \beta \geq 0$. Then, for any routing decision $a \in \{P, R, A\}$, and higher-order prediction $\pi \in \Delta\Delta\mathcal{Y}$ in the range of F, we are guaranteed that*

$$\left| \mathop{\mathbb{E}}_{x\sim\mathcal{D}_\mathcal{X}}[\mathsf{Cost}(x, a; T)|F(x) = \pi] - \widehat{\mathsf{Cost}}_F(x, a; T) \right| \leq B\epsilon/2.$$

We first prove the theorem using the lemma, and then present the proof of the lemma.

Proof of Theorem 3.1. We define our router r to take the action that minimizes the simulated routing loss computed using F (Definition D.1).

$$r(x, T) := \arg\min_{a\in\{P,R,A\}} \widehat{\mathsf{Cost}}_F(x, a; T).$$

Note that this implies r's routing decisions are constant over each level set of the higher-order predictor (and equivalently f).

Leveraging Lemma D.2, we now show that these actions are also good minimizers for the true routing loss, on average over partitions. Let $\pi \in \Delta\Delta\mathcal{Y}$ be some value in the range of F, and fix a configuration $T = (L, \alpha, \beta)$ where L is a B-bounded proper loss and $\alpha, \beta \geq 0$. Note that $f(x)$ is constant for all points x with $F(x) = \pi$ because the two predictors have the same level sets. We denote this constant value as $f(x) = p$. We bound the routing loss over this level set as follows:

$$\begin{aligned}
&\mathop{\mathbb{E}}_{x\sim\mathcal{D}_\mathcal{X}}[\mathsf{Cost}(x, r(x, T); T)|F(x) = \pi] \\
&\leq \widehat{\mathsf{Cost}}_F(x, r(x, T); T) + B\epsilon/2 && \text{(Lemma D.2)} \\
&\leq \widehat{\mathsf{Cost}}_F(x, r_f(x, T); T) + B\epsilon/2 && \text{(Definition of } r(x, T)) \\
&\leq \mathop{\mathbb{E}}_{x\sim\mathcal{D}_\mathcal{X}}[\mathsf{Cost}(x, r_f(x, T); T)|F(x) = \pi] + B\epsilon && \text{(Lemma D.2)}
\end{aligned}$$

Thus, taking the average over all level sets of the higher-order predictor, we conclude that the excess routing cost is bounded:

$$\mathop{\mathbb{E}}_{x\sim\mathcal{D}_\mathcal{X}}[\mathsf{Cost}(x, r(x, T); T)] - \mathop{\mathbb{E}}_{x\sim\mathcal{D}_\mathcal{X}}[\mathsf{Cost}(x, r_f(x, T); T)] \leq B\epsilon.$$

Because this holds for any configuration T satisfying the conditions of the theorem, the proof is complete. □

We now finish up by proving our helper lemma.

Proof of Lemma D.2. It suffices to prove the statement for an arbitrary $T = (L, \alpha, \beta)$ and predicted mixture π, where L is a B-bounded proper loss, and $\alpha, \beta \geq 0$. Let F^* denote the distribution over $f^*(x)$ induced by drawing $x \sim \mathcal{D}_\mathcal{X}$ conditioned on $F(x) = \pi$. Note that by our assumption that F has the same level sets as f, $f(x)$ is constant for all points with $F(x) = \pi$. We denote this constant value $f(x) = p$.

We bound the maximum difference in real vs simulated routing costs by taking a maximum over all possible routing decisions. We use $\widehat{p} \in \Delta\mathcal{Y}$ to denote an independent draw from the mixture π:

$$\begin{aligned}
&\max_{a\in\{P,R,A\}} \left| \mathop{\mathbb{E}}_{x\sim\mathcal{D}_\mathcal{X}}[\mathsf{Cost}(x, a; T)|F(x) = \pi] - \widehat{\mathsf{Cost}}_F(x, a; T) \right| \\
&= \max\{ \left| \mathop{\mathbb{E}}_{p^*\sim F^*}[L(p^*, p)] - \mathop{\mathbb{E}}_{\widehat{p}\sim\pi}[L(\widehat{p}, p)] \right|, \left| \mathop{\mathbb{E}}_{p^*\sim F^*}[L(p^*, p^*)] - \mathop{\mathbb{E}}_{\widehat{p}\sim\pi}[L(\widehat{p}, \widehat{p})] \right|, 0\}
\end{aligned}$$

We use the following dual definition of the 1-Wasserstein distance. Let $\mathcal{G}_{Lip}$ be the class of all 1-Lipschitz functions $g : \Delta\mathcal{Y} \to \mathbb{R}$. We can equivalently write the Wasserstein distance between $\pi_1 \in \Delta\Delta\mathcal{Y}$ and $\pi_2 \in \Delta\Delta\mathcal{Y}$ as

$$W_1(\pi_1, \pi_2) = \max_{g\in\mathcal{G}_{Lip}} \mathop{\mathbb{E}}_{p_1\sim\pi_1}[g(p_1)] - \mathop{\mathbb{E}}_{p_2\sim\pi_2}[g(p_2)].$$

We note that by Lemma B.2, the functions $g_{TL}(p^*) = L(p^*, p)$ and $g_{IR}(p^*) = L(p^*, p^*)$ both satisfy $B/2$-Lipschitzness, and thus we can further bound our maximum by a multiplicative factor of the maximum over all 1-Lipschitz functions:

$$\frac{B}{2} \max_{g \in \mathcal{G}_{Lip}} \mathop{\mathbb{E}}_{p^* \sim F^*}[g(p^*)] - \mathop{\mathbb{E}}_{\widehat{p} \sim \pi}[g(\widehat{p})] = \frac{B}{2} W_1(F^*, \pi) \qquad \text{(Wasserstein duality)}$$

$$\leq B\epsilon/2 \qquad \text{(Higher-order calibration guarantee)}$$

Thus, we conclude that the difference between the true and estimated routing loss values is at most $B\epsilon/2$. □

D.2. Generalizing to Additional Oracles

While we state our main result for a particular routing setting where the oracle makes Bayes-optimal predictions, our proof techniques can naturally be extended to settings where we have access to multiple oracles, each with varying cost. The key requirement we make is that each oracle's loss at a point x can be expressed as a known function of $f^*(x)$. We call such an oracle a *Bayes-dependent oracle*.

Definition D.3 (Bayes-Dependent Oracle)**.** An oracle $g_O(x) : \mathcal{X} \to \Delta\mathcal{Y}$ is *Bayes-Dependent* if there exists a function $c : \mathcal{L} \times \Delta\mathcal{Y} \to \mathbb{R}$ such that for any point $x \in \mathcal{X}$, $L(f^*(x), g_O(x)) = c(L, f^*(x))$, i.e. the expected loss of the oracle at x can be expressed purely as a function of $f^*(x)$ and the loss of interest.

We note that this definition captures a wide range of oracles that we might have access to in practice. As one example, a common alternative option to prediction might be to ask one or more human experts for their opinion on how to classify a particular point. Each human expert's opinion can be viewed as a draw $y \sim f^*(x)$ from the true conditional distribution.

Thus, the expected loss of querying k human experts and combining their opinions using some aggregation function $a : \mathcal{Y}^k \to \Delta\mathcal{Y}$ can be expressed as a function of $f^*(x)$:

$$c(L, f^*(x)) = \mathop{\mathbb{E}}_{y_1, \ldots, y_k \sim f^*(x)}[L(f^*(x), a(y_1, ..., y_k))].$$

We can thus expand our routing setting to assume that we have access to k oracles whose losses are computed by (known) functions $c_1, ..., c_k$. Our action set also expands to $\{P, A, R_1, ..., R_k\}$, where R_i denotes routing to the ith oracle, and we have k routing penalties $\alpha_1, ..., \alpha_k \geq 0$ in addition to the abstention penalty $\beta \geq 0$.

In such a setting, we can naturally derive a generalized version of Lemma D.2, where the simulated costs for action R_i are now computed as

$$\widehat{\mathsf{Cost}}_F(x, R_i; T) := \mathop{\mathbb{E}}_{p^* \sim F(x)}[c_i(L, p^*)] + \alpha_i.$$

Lemma D.4. *Let F be a ϵ-higher-order-calibrated predictor whose level sets match those of f and let $T = (L, \alpha_{1:k}, \beta)$ be a routing configuration where L is a B-bounded proper loss $L : \Delta\mathcal{Y} \times \Delta\mathcal{Y} \to [0, B]$, and $\alpha_{1:k}, \beta \geq 0$. Let each $c_i(L, \cdot)$ satisfy γ-Lipschitz continuity.*

Then, for any routing decision $a \in \{P, R_1, ..., R_k, A\}$, and higher-order prediction $\pi \in \Delta\Delta\mathcal{Y}$ in the range of F, we are guaranteed that

$$\left| \mathop{\mathbb{E}}_{x \sim \mathcal{D}_{\mathcal{X}}}[\mathsf{Cost}(x, a; T) | F(x) = \pi] - \widehat{\mathsf{Cost}}_F(x, a; T) \right| \leq \max\{B/2, \gamma\}\epsilon.$$

The proof follows an almost identical argument to Lemma D.2, which we present below. With this lemma in hand, we can apply it to get a generalized version of our main result, Theorem 3.1. We present only the statement without proof, as the proof is identical to the proof of Theorem 3.1, plugging in the updated bound of $\max\{B/2, \gamma\}\epsilon$ on the gap between true and simulated routing costs.

Theorem D.5 (Prediction-Optimal Routing with Bayes-Dependent Oracles (Generalization of Theorem 3.1))**.** *Suppose we have an ϵ-higher-order calibrated predictor F whose level sets match those of f (i.e., F can be viewed as a function of f). Then we can construct a flexible router r based on F with the following near-optimality property. For any routing*

configuration $T = (L, \alpha_{1:k}, \beta)$ where L is a B-bounded proper loss, $\alpha_{1:k}, \beta \geq 0$, and each routing decision R_i routes to a Bayes-dependent oracle whose loss can be computed using the γ-Lipschitz function $c_i(L, \cdot)$, then we have

$$\mathop{\mathbb{E}}_{x\sim\mathcal{D}_\mathcal{X}}[\mathsf{Cost}(x, r(x,T); T) - \mathsf{Cost}(x, r_f(x,T); T)] \leq \max\{B\epsilon, 2\gamma\epsilon\}.$$

We now provide the proof of Lemma D.4.

Proof of Lemma D.4. It suffices to prove the statement for an arbitrary $T = (L, \alpha_{1:k}, \beta)$ and predicted mixture π, where L is a B-bounded proper loss, and $\alpha_{1:k}, \beta \geq 0$. Let F^* denote the distribution over $f^*(x)$ induced by drawing $x \sim \mathcal{D}_\mathcal{X}$ conditioned on $F(x) = \pi$. Note that by our assumption that F has the same level sets as f, $f(x)$ is constant for all points with $F(x) = \pi$. We denote this constant value $f(x) = p$.

We bound the maximum difference in real vs simulated routing costs by taking a maximum over all possible routing decisions:

$$\begin{aligned}
&\max_{a\in\{P,R_{1:k},A\}} \left| \mathop{\mathbb{E}}_{x\sim\mathcal{D}_\mathcal{X}}[\mathsf{Cost}(x,a;T)|F(x)=\pi] - \widehat{\mathsf{Cost}}_F(x,a;T) \right| \\
&= \max\{\left| \mathop{\mathbb{E}}_{p^*\sim F^*}[L(p^*,p)] - \mathop{\mathbb{E}}_{\widehat{p}\sim\pi}[L(\widehat{p},p)] \right|, 0, \max_{i\in[k]}\{\left| \mathop{\mathbb{E}}_{p^*\sim F^*}[c_i(L,p^*)] - \mathop{\mathbb{E}}_{\widehat{p}\sim\pi}[c_i(L,\widehat{p})] \right|\}\}
\end{aligned}$$

We use the following dual definition of the 1-Wasserstein distance. Let $\mathcal{G}_{Lip}$ be the class of all 1-Lipschitz functions $g : \Delta\mathcal{Y} \to \mathbb{R}$. We can equivalently write the Wasserstein distance between $\pi_1 \in \Delta\Delta\mathcal{Y}$ and $\pi_2 \in \Delta\Delta\mathcal{Y}$ as

$$W_1(\pi_1, \pi_2) = \max_{g\in\mathcal{G}_{Lip}} \mathop{\mathbb{E}}_{p_1\sim\pi_1}[g(p_1)] - \mathop{\mathbb{E}}_{p_2\sim\pi_2}[g(p_2)].$$

We note that by Lemma B.2, the function $g_{TL}(p^*) = L(p^*, p)$ satisfies $B/2$-Lipschitzness, and by assumption each $c_i(L, \cdot)$ satisfies γ-Lipschitz-ness, and thus we can further bound our maximum by a multiplicative factor of the maximum over all 1-Lipschitz functions:

$$\begin{aligned}
\max\{\gamma, \frac{B}{2}\} \max_{g\in\mathcal{G}_{Lip}} \mathop{\mathbb{E}}_{p^*\sim F^*}[g(p^*)] - \mathop{\mathbb{E}}_{\widehat{p}\sim\pi}[g(\widehat{p})] &= \max\{\frac{B}{2}, \gamma\} W_1(F^*, \pi) && \text{(Wasserstein duality)} \\
&\leq \max\{B/2, \gamma\}\epsilon && \text{(Higher-order calibration guarantee)}
\end{aligned}$$

Thus, we conclude that the difference between the true and estimated routing loss values is at most $\max\{B/2, \gamma\}\epsilon$. □

E. Characterizing Useful Partitions

As discussed in Section 4, the choice of partition $\mathcal{P}$ is critical to the success of our method. Even among all partitions that allow for sample-efficient estimation of the loss decomposition, some are better than others. The following lemma gives a property of a partition that characterizes how the best partition-dependent decisions compare to the pointwise-optimal decisions: partitions that group points with similar amounts of reducible loss have routing performance very close to the pointwise-optimal cost.

For simplicity, we focus on the setting where $\beta = \infty$, i.e. the routing policy need only choose between prediction and routing to the oracle. However, using similar techniques an expression can be derived for general β, and is a more complicated expression capturing the variance in both the reducible and irreducible loss within each partition cell.

Lemma E.1. *Given a partition $\mathcal{P}$ of the space $\mathcal{X}$ and a routing configuration $T = (L, \alpha, \beta)$ for a proper loss L and $\alpha \geq 0$, $\beta = \infty$ (no abstention), consider a bin within the partition, $P \in \mathcal{P}$. Denote RL as the random variable corresponding to the reducible loss of points $x \sim \mathcal{D}_\mathcal{X}$, conditioned on falling within P, with mean $\overline{\mathrm{RL}}$.*

Then, the excess routing cost incurred by picking the best single routing decision for all of P compared to the pointwise-optimal router has the following upper bound:

$$\min_{a\in\{P,R,A\}} \mathop{\mathbb{E}}_{x\sim\mathcal{D}_\mathcal{X}}[\mathsf{Cost}(x,a;T)|x\in P] - \mathop{\mathbb{E}}_{x\sim\mathcal{D}_\mathcal{X}}[\mathsf{Cost}(x,r^*(x,T);T)|x\in P] \leq \frac{1}{2}\,\mathbb{E}[\left|\mathrm{RL} - \overline{\mathrm{RL}}\right|].$$

We note that this bound is tight in that there exists an α that results in excess cost matching the upper bound.

Proof. Similar to RL, let IL denote the random variable corresponding to the irreducible loss for a point $x \sim \mathcal{D}_{\mathcal{X}}$ conditioned on $x \in P$, with mean $\overline{\mathrm{IL}}$.

Substituting these values into our definition of Cost, and noting that because $\beta = \infty$, it is never optimal to abstain, the excess cost expression simplifies to

$$\begin{aligned}\min\{\overline{\mathrm{IL}} + \overline{\mathrm{RL}}, \overline{\mathrm{IL}} + \alpha\} - \mathbb{E}[\min\{\mathrm{IL} + \mathrm{RL}, \mathrm{IL} + \alpha\}] &= \min\{\overline{\mathrm{IL}} + \overline{\mathrm{RL}}, \overline{\mathrm{IL}} + \alpha\} - \mathbb{E}[\mathrm{IL} + \min\{\mathrm{RL}, \alpha\}] \\ &= \min\{\overline{\mathrm{RL}}, \alpha\} - \mathbb{E}[\min\{\mathrm{RL}, \alpha\}]].\end{aligned}$$

When $\alpha \leq \overline{\mathrm{RL}}$, this expression is increasing with α, and when $\alpha \geq \overline{\mathrm{RL}}$, the expression is non-increasing as α increases. Thus, we conclude that $\alpha^* = \overline{\mathrm{RL}}$ maximizes the expression. We substitute in α^* to get an upper bound on the excess cost, which simplifies the excess cost to

$$\begin{aligned}\overline{\mathrm{RL}} - \mathbb{E}[\min\{\mathrm{RL}, \overline{\mathrm{RL}}\}] &= \mathbb{E}[(\overline{\mathrm{RL}} - \mathrm{RL})^+] \\ &= \frac{1}{2}\,\mathbb{E}[\left|\overline{\mathrm{RL}} - \mathrm{RL}\right|].\end{aligned}$$

□

E.1. Bucketing Strategies

With this intuition in mind, we recommend the following strategies for partitioning:

E.1.1. Top-Class Binning

Calibrating on the full vector of probability outputs is intractable for multiclass problems due to the exponential growth of the output space. To maintain sample-efficiency, we partition the space based on the *top-class prediction* and its associated confidence (e.g., "Class 5, Confidence 0.9-1.0"). This ensures the partition size scales linearly with the number of classes, making calibration more sample-efficient.

E.1.2. Centroid Calibration

In the "Optional Re-Calibration Step," we post-process the weak predictor to output the centroid of the empirical distribution of true labels in its assigned bin. This alignment ensures the predictor is constant over each level set of the higher-order predictor, satisfying the assumptions of Theorem 3.1. In practice, we found this not only stabilizes routing but often reduces the weak model's original prediction loss, as it is essentially a form of (first-order) post-hoc calibration.

Remark E.2. We note that if the "weak" model is already highly performant, collapsing its output to the centroids of a coarse partition may discard useful signal. In such cases, this step should be skipped.

E.1.3. Feature-based Binning

Finally, relying solely on the weak model's confidence may limit a partition's usefulness. In practice, it may be useful to additionally define bins using auxiliary features that can be expected to correlate with the amount of irreducible uncertainty, such as a heuristic for the blurriness of an image in an image classification task. We provide a synthetic example of how feature-based binning can improve over binning by prediction value in Figure 7.

F. Additional Experiment Discussion

F.1. Dataset Details

We detail each dataset and associated weak predictor architecture below. We do not perform any hparam optimization since we are not trying to obtain the best performing model in any setting, we mostly use defaults from other implementations.

- **Synthetic Data**: A set of synthetic data functions with baseline aleatoric uncertainty inspired by Johnson et al. (2024). We give further explanation of how we generate this data in Section F.2. For the weak predictor, we train a simple

ResNet with three residual blocks and two MLP layers in each block, following the architecture used by Johnson et al. (2024). The model achieves nearly perfect fit to the ground-truth function, since we can generate infinite synthetic examples for training.

- **CIFAR-10H**: A re-annotation of the CIFAR-10 test set where each image has at least 47 human labels (Peterson et al., 2019). As the weak predictor, we train a standard ResNet34 (He et al., 2016) on the original CIFAR-10 training split. Our model achieves 87% accuracy on the original CIFAR-10 test split, consistent with being a weak, imperfect predictor. We split the 10K example CIFAR-10H test-set into 5K for post-hoc higher order calibration, and 5K hold-out examples for testing the router. Dataset License: CC BY-NC-SA 4.0 Further hparams:
  - **Training epochs:** 50.
  - **Batch size:** 512.
  - **Optimizer:** AdamW.
  - **Maximum learning rate:** 3.799×10^{-3}.
  - **Weight decay:** 3.656×10^{-1}.
  - **AdamW defaults:** $\beta_1 = 0.9$, $\beta_2 = 0.999$, $\epsilon = 10^{-8}$.
  - **Learning-rate schedule:** OneCycleLR stepped once per minibatch, with 50 epochs and $\texttt{steps_per_epoch} = |\mathcal{D}_{\text{train}}|/512$.
  - **Loss:** cross-entropy.
  - **Gradient clipping:** value clipping at 0.1.

- **SNLI**: Natural language inference task, classify pairs of sentences into "contradiction", "neutral", or "entailment". Five human labels per example (Bowman et al., 2015). We finetune a 100M parameter Bert model (Devlin et al., 2018) (License: apache-2.0) on the classification task given pairs of sentences, using the majority label. Training is done on the original SNLI train set. The resulting test accuracy of the weak predictor is 85%. We run routing experiments by splitting the SNLI 10K test set, with 5K examples used for post-hoc higher order calibration, and 5K hold-out examples used for testing the router. Dataset License: CC BY-SA 4.0. Further hparams:
  - **SNLI model:** `bert-base-uncased` sequence classifier with 3 output labels.
  - **Training epochs:** 3.
  - **Batch size:** 32.
  - **Optimizer:** AdamW.
  - **Learning rate:** 2×10^{-5}.
  - **AdamW defaults:** $\beta_1 = 0.9$, $\beta_2 = 0.999$, $\epsilon = 10^{-8}$, weight decay 0.01.
  - **Learning-rate schedule:** linear decay with warmup.
  - **Warmup:** 10% of total training steps.
  - **Loss:** cross-entropy via the HuggingFace sequence-classification loss.
  - **Gradient clipping:** norm clipping at 1.0.
  - **Maximum sequence length:** 128 tokens.

- **ChaosNLI**: Re-labeling of SNLI and MNLI (Williams et al., 2018) with 100 human labels per sentence pair (Nie et al., 2020b), increasing the amount of aleatoric uncertainty. As the weak predictor, we utilize the original SNLI/MNLI predictions given by the authors (Nie et al., 2020b). These were generated by finetuning a RoBERTa-large model on a variety of NLI tasks (Nie et al., 2020a). We use a 3113 subset of the ChaosNLI dataset belonging to SNLI and MNLI, and further use half the data for post-hoc model higher-order calibration, and the other half for routing experiments. Dataset License: CC BY-NC 4.0

All predictions for routing were generated in <1 hour on a single A100 GPU. Running routing algorithms themselves is cheap and CPU-bound.

F.2. Synthetic Data Generation Details

We employ a synthetic binary classification setting to precisely control the learning environment, particularly the irreducible uncertainty. The inputs are real-valued scalars drawn from a standard normal distribution, $x \sim \mathcal{N}(0, 1)$. We define the target distribution via an explicit Bayes-optimal predictor $f^* : \mathbb{R} \to [0, 1]$, which maps each input x to the true conditional probability $P(y = 1 \mid x)$. By explicitly designing f^*, we can directly manipulate both the magnitude and the variance of the irreducible uncertainty across the input space.

For a given f^*, we train the weak predictor on a dataset of 10,000 samples (x, y), where $y \sim \mathsf{Ber}(f^*(x))$. To perform higher-order calibration, we generate a k-snapshot calibration set containing 5,000 instances. For each input x, we sample $k = 100$ independent and identically distributed labels $y_i \sim \mathsf{Ber}(f^*(x))$. Finally, we evaluate all methods on a held-out test set of 500,000 samples.

Figure 5 visualizes the three target distributions (f^*) we consider, the predictions of the trained weak models, and the post-hoc calibrated predictions used in our experiments. The precise functions used to construct each of these examples is detailed below:

- Figure 5a:

$$f^*(x) = \frac{0.98u(x) + 1}{2}$$

  where the intermediate variables $u(x)$, $v(x)$, and $w(x)$ are defined as:

$$u(x) = 0.6\cos(v(x)) + 0.4\cos(4.2x)$$

$$v(x) = \text{sgn}(x)\,(120|x| - 112w(x) - 0.0635)$$

$$w(x) = 0.2\ln\left(1 + \exp\left(\frac{|x| - 1}{0.2}\right)\right)$$

- Figure 5b:

$$f^*(x) = \begin{cases} 0 & \text{if } x \leq -1 \\ \frac{\sin(100x)}{2} + 0.5 & \text{if } -1 < x < 1 \\ 1 & \text{if } x \geq 1 \end{cases}$$

- Figure 5c:

$$f^*(x) = \begin{cases} 0.5 & \text{if } x \leq -1 \\ \frac{\sin(100x)}{4} + 0.5 & \text{if } -1 < x \leq -0.5 \\ 0.25 & \text{if } -0.5 < x \leq 0 \\ \frac{\sin(100x)}{4} + 0.5 & \text{if } 0 < x \leq 0.5 \\ \frac{\sin(100x)}{4} + 0.25 & \text{if } x > 0.5 \end{cases}$$

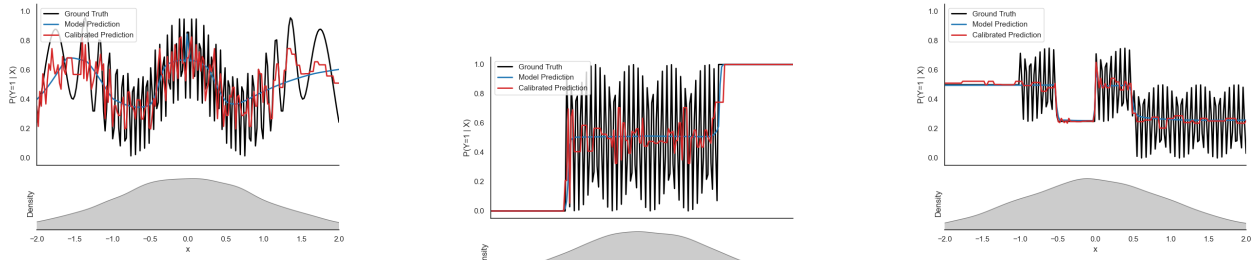
F.3. Benefits of Decomposition

F.3.1. Further Discussion of Decomposition

We highlight two key factors that dictate the benefits of our method, and more broadly any method that can separate irreducible and reducible loss:

Reducible vs. Total Loss Correlation In environments where reducible and total loss are highly correlated, a ranking based on total uncertainty serves as a near-perfect proxy for a ranking by reducible loss. In such cases, simpler confidence-based metrics are sufficient for near-optimal routing. We note that this correlation may be an inherent property of the dataset, such as if all points have a similar amount of irreducible loss, or can be a byproduct of the training dynamics used to obtain the weak predictor.

We provide an example of the latter case in Figure 6. While the target function (Figure 5b) has significant variance in the amount of irreducible uncertainty across inputs, the highly varied portion of the data (where the model experiences the



(a) Sinusoidal ground truth with variable irreducible uncertainty, matching the test function used in the synthetic experiments of Johnson et al..

(b) Step function distribution with varying amounts of irreducible uncertainty.

(c) Mixed target distribution where regions of low and high variance in the amount of irreducible uncertainty share the same mean.

Figure 5. Examples of synthetic binary data settings used in experiments, as described in Section F.2. The marginal distribution over x is the standard normal distribution, and the black curve represents the true conditional expectation of y for each x, i.e. $f^*(x)$. The blue curve captures the predictions of a model trained on the data, and the red curve captures the predictions of the model after post-hoc calibration on the same buckets used by our routing framework.

most irreducible loss) corresponds with the places where a weak model is likely to predict a value close to 0.5, the maximal amount of total uncertainty, causing routing based on total uncertainty to perform almost as well as routing based on the reducible loss.

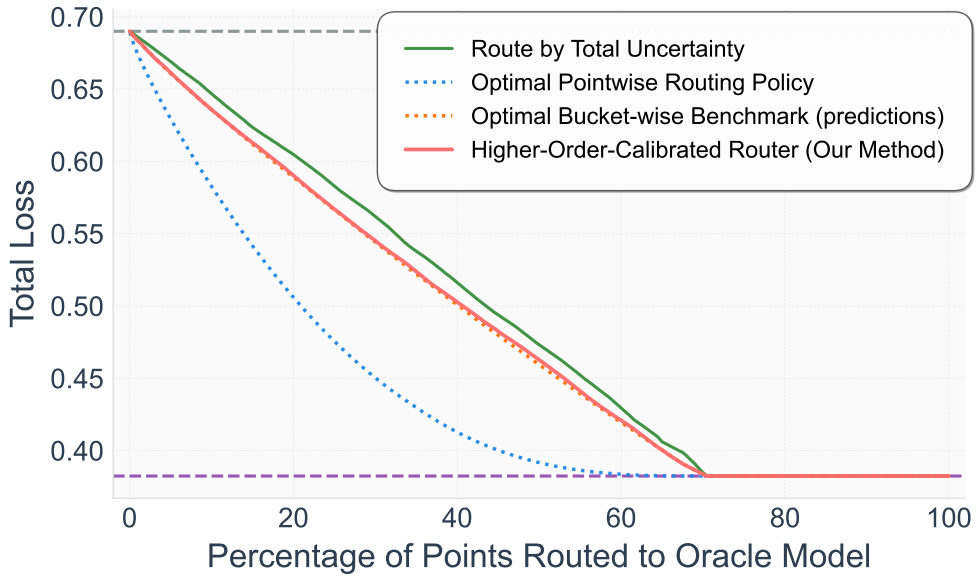


Figure 6. The above routing curve depicts the performance of our routing framework on the synthetic target function visualized in Figure 5b. We highlight this as an example of a setting in which there is a large amount of variation in the irreducible uncertainty of the data, but training dynamics result in a model that doesn't significantly benefit from decomposition, as evidenced by the fact that the performance achieved by a total uncertainty-based approach is closely aligned with the routing curve of our higher-order method.

The Impact of Bucket Structure As discussed in Section 4, for bucket-based approaches like ours, the choice of partition is critical. Even if reducible and total loss are uncorrelated in a pointwise sense, a poorly chosen bucketing scheme may average these values in a way that induces a high correlation per-bucket. To illustrate this, Figure 7 provides a synthetic example where introducing even a small number of feature-based buckets results in significant performance gains from decomposition compared to bucketing only by level sets.

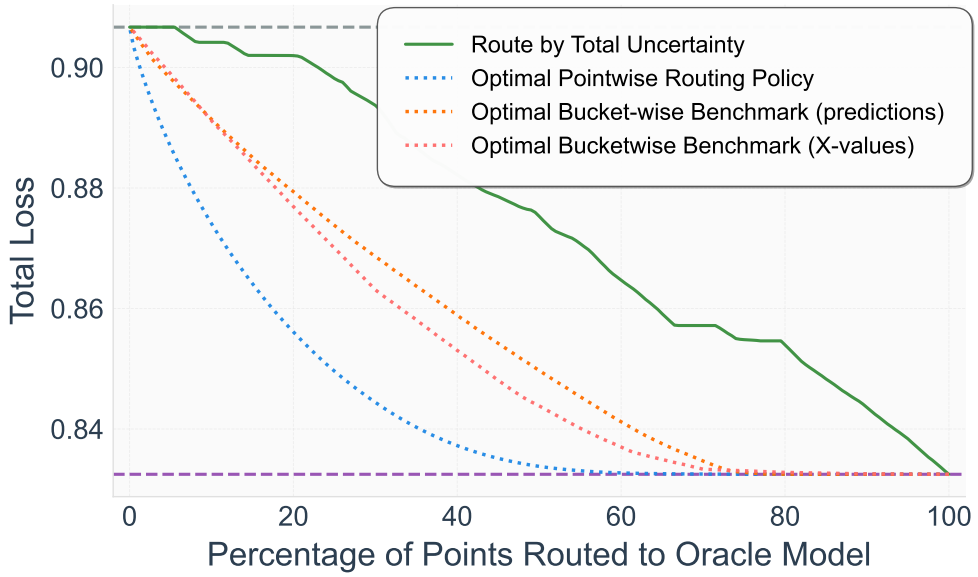
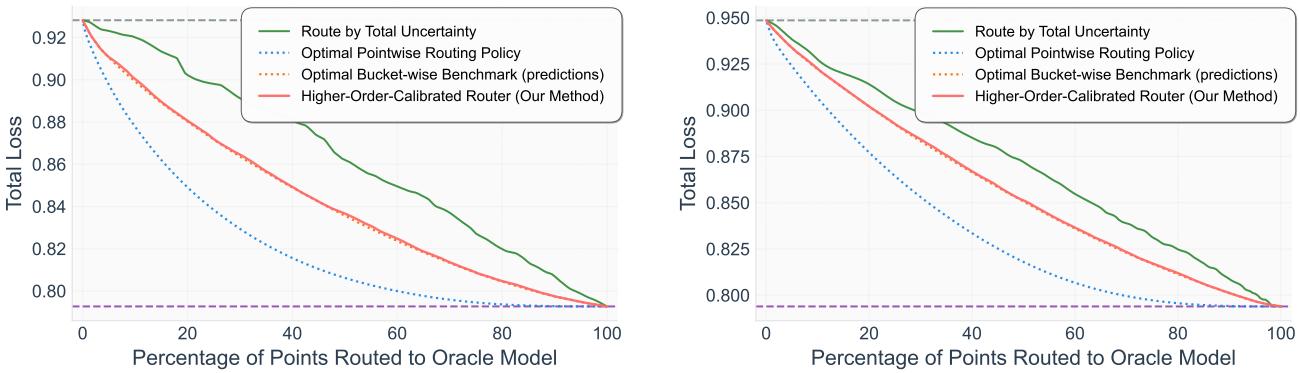


Figure 7. The above routing curve compares the performance of two bucketing approaches on synthetic data with target function depicted in Figure 5c. The orange and pink lines both depict the optimal performance of a router that must be constant over a particular partition of the input space of buckets. The orange line is the performance when 30 buckets of equal size are created based on the level sets (predictions) of the weak model, while the pink line results from using just 10 equal-sized buckets that split on the input value x. The improved performance of the pink line illustrates how the particular choice of bucketing can be critical the performance of our routing framework.

G. Routing Curve Plots

This section provides the full routing curve results for all evaluated datasets, comparing the calibrated and uncalibrated models.

G.1. Synthetic Data

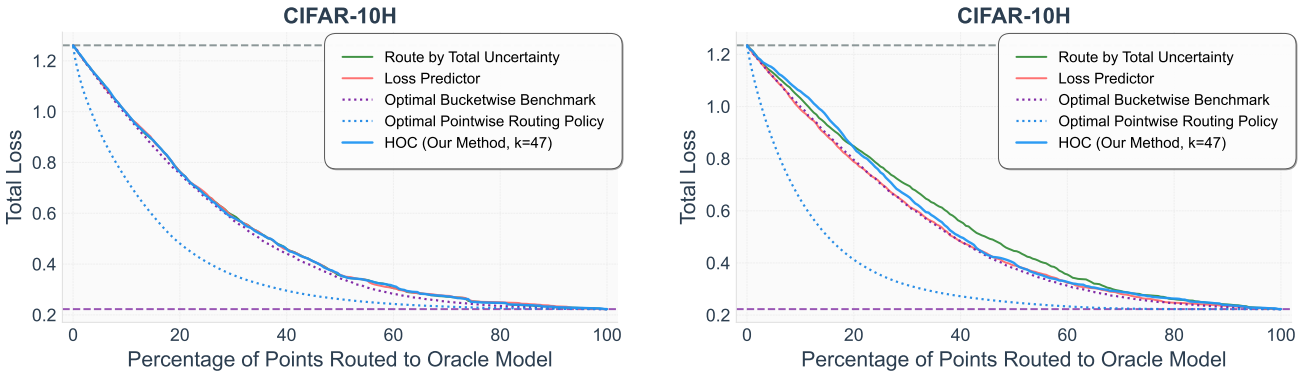


(a) Calibrated Weak Model

(b) Uncalibrated Weak Model

Figure 8. Routing curves for synthetic data as described in Section F.2 and target function as depicted in Figure 5a. The right (b) employs an out-of-the-box weak predictor for the dataset, whereas the left (a) uses the same predictor but with an additional post-hoc calibration step on the same buckets used by the higher-order-calibrated router.

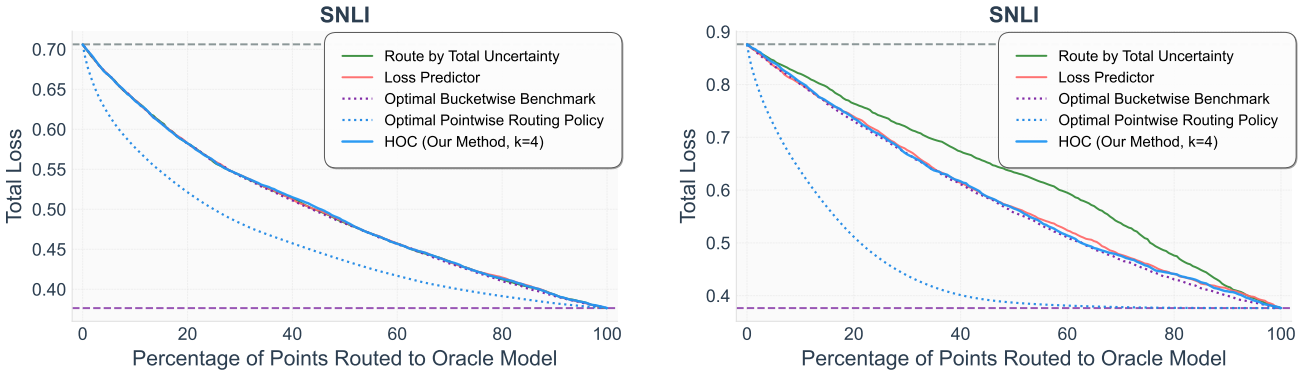
G.2. CIFAR-10H



(a) Calibrated Weak Model *(b)* Uncalibrated Weak Model

Figure 9. Routing curves for the CIFAR-10H dataset. The right (b) employs an out-of-the-box weak predictor for the dataset, whereas the left (a) uses the same predictor but with an additional post-hoc calibration step on the same buckets used by the higher-order-calibrated router.

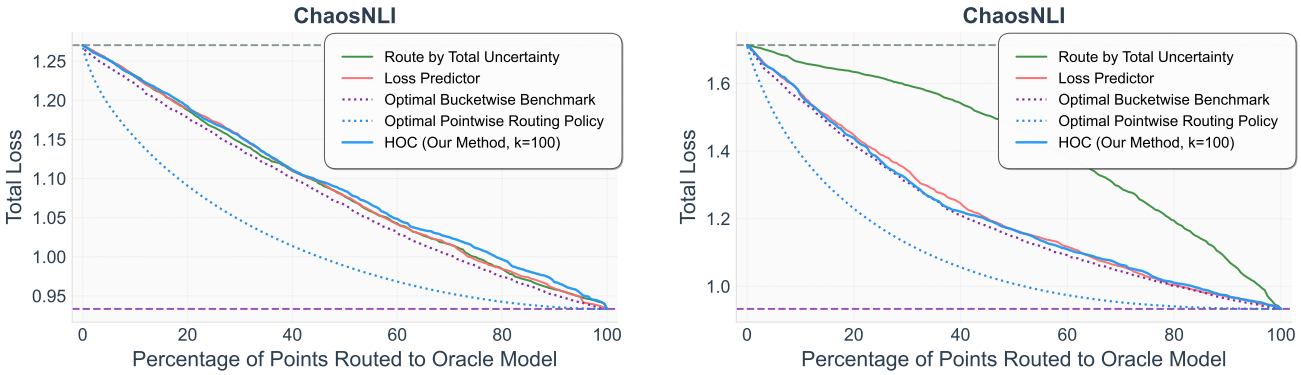
G.3. SNLI



(a) Calibrated Weak Model *(b)* Uncalibrated Weak Model

Figure 10. Routing curves for the SNLI dataset. The right (b) employs an out-of-the-box weak predictor for the dataset, whereas the left (a) uses the same predictor but with an additional post-hoc calibration step on the same buckets used by the higher-order-calibrated router.

G.4. ChaosNLI



(a) Calibrated Weak Model

(b) Uncalibrated Weak Model

Figure 11. Routing curves for the ChaosNLI dataset. The right (b) employs an out-of-the-box weak predictor for the dataset, whereas the left (a) uses the same predictor but with an additional post-hoc calibration step on the same buckets used by the higher-order-calibrated router.

H. Loss Flexibility Plots

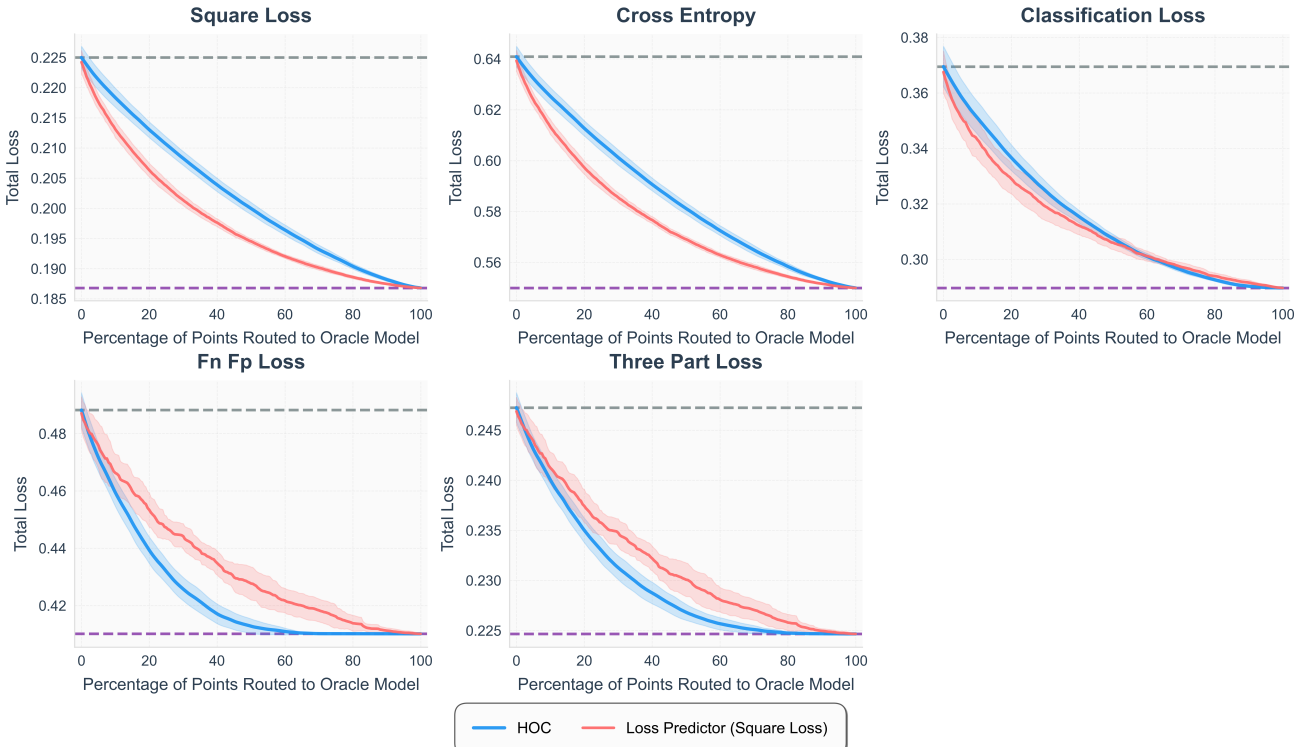
H.1. Explanation of Loss Functions

We overview the different losses appearing in the comparison plots in this section.

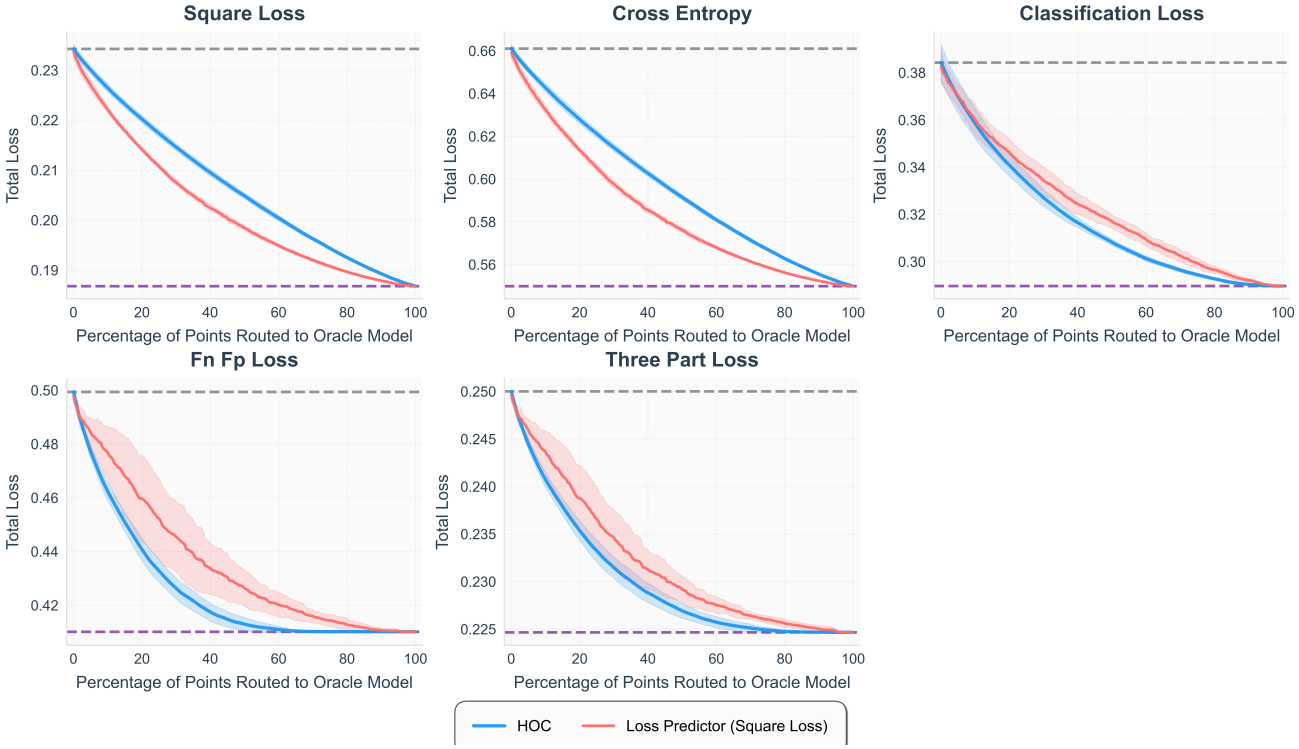
Remark on Proper Losses: Our routing framework can flexibly adapt to various new losses, with the restriction that these losses must be proper (see Section 2.2 for a definition). While standard functions like Brier or Cross-Entropy are (strictly) proper out-of-the-box, other losses such as the classification loss are not, and should not be evaluated directly on raw probability predictions. To handle this, some of the losses below couple the underlying decision cost with an optimal post-processing step (such as reweighting or thresholding the predicted probabilities p). This ensures that the expected loss minimized by the model continues to act as a proper loss with respect to the original predicted probabilities.

- Square (Brier) Loss: This denotes the typical squared distance between the label distribution and prediction, $\ell(y, p) = \|y - p\|_2^2$.
- Cross Entropy Loss: This is the standard negative log-likelihood, defined as $\ell(y, p) = -\sum_c y_c \log(p_c)$.
- Classification Loss: Let $\widehat{y} = \arg\max_c p_c$ be the predicted class with maximal probability. The expected 0-1 error given the true probabilities y is $\ell(y, p) = 1 - y_{\widehat{y}}$.
- Weighted False-Positive Loss ("Fn Fp Loss", Binary only): This applies asymmetric penalties to false positives (c_{FP}) and false negatives (c_{FN}). The decision $\widehat{y} \in \{0, 1\}$ is determined by thresholding the predicted odds: $\widehat{y} = \mathbf{1}[\frac{p_1}{p_0} \geq \frac{c_{FP}}{c_{FN}}]$. The expected loss is evaluated against the true probabilities y as $\ell(y, p) = c_{FP} \cdot \mathbf{1}[\widehat{y} = 1]y_0 + c_{FN} \cdot \mathbf{1}[\widehat{y} = 0]y_1$.
- Three Part Loss (binary only): This loss extends the weighted false positive loss by introducing an alternative third action with a fixed cost. Let p_1 be the predicted probability of the positive class. The loss is defined conditionally: $\ell(y, p) = y_1$ if $p_1 < 0.25$; a constant 0.25 if $0.25 \leq p_1 < \frac{15}{16}$; and $4y_0$ if $p_1 \geq \frac{15}{16}$.
- Asymmetric Class Penalty (multiclass): This applies a penalty multiplier γ strictly when predicting class 0. The decision scores s are biased against class 0 such that $s_0 = \gamma p_0 + (1 - \gamma)$, while $s_c = p_c$ for $c \neq 0$. For the optimal action $\widehat{y} = \arg\max_c s_c$, the loss is $\gamma(1 - y_0)$ if $\widehat{y} = 0$, and standard $1 - y_{\widehat{y}}$ for all other classes.

H.2. Synthetic Data



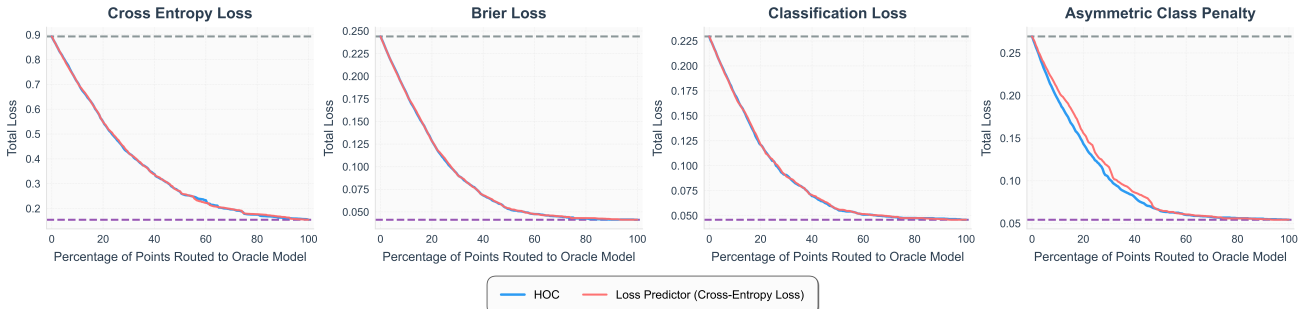
(a) Calibrated Weak Model



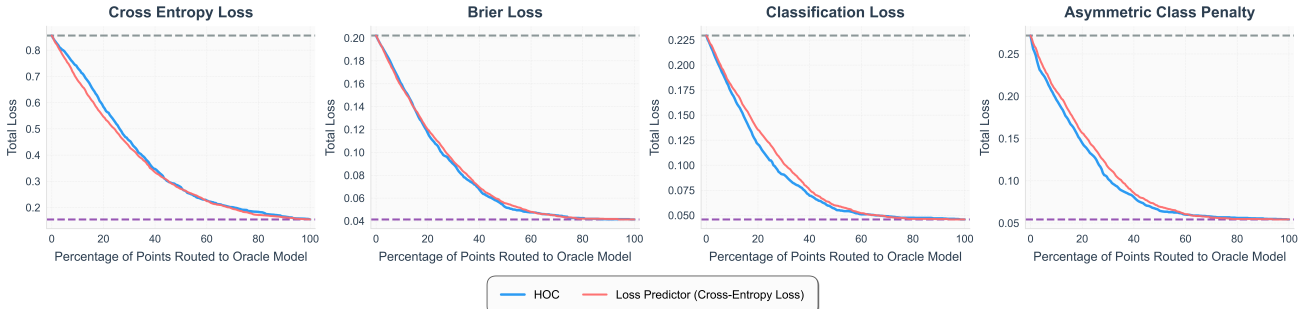
(b) Uncalibrated Weak Model

Figure 12. Routing performances for different target loss functions for synthetic data as described in Section F.2 and target function as depicted in Figure 5a. The loss predictor baseline is a static loss predictor trained to predict the amount of reducible square loss. The right (b) employs an out-of-the-box weak predictor for the dataset, whereas the left (a) uses the same predictor but with an additional post-hoc calibration step on the same buckets used by the higher-order-calibrated router. Error bars show the standard deviation over 10 runs, re-randomizing both the training set used to train the weak model and the calibration set used to higher-order calibrate and train a loss predictor.

H.3. CIFAR-10H



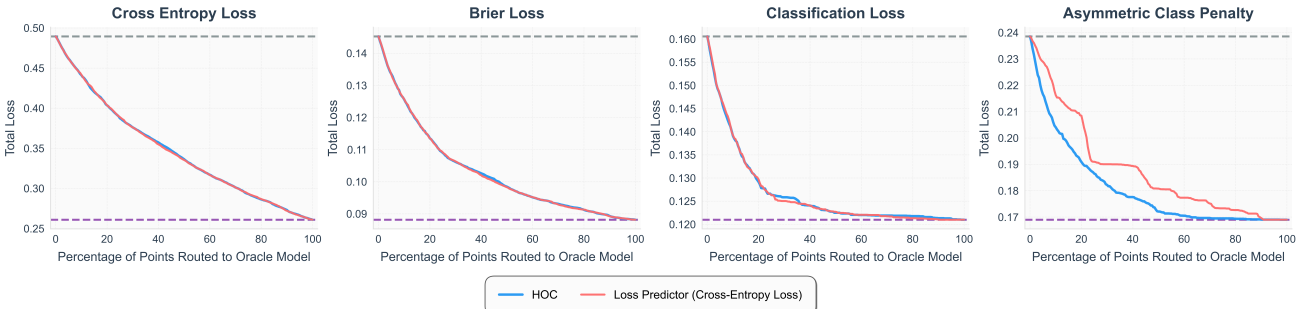
(a) Calibrated Weak Model



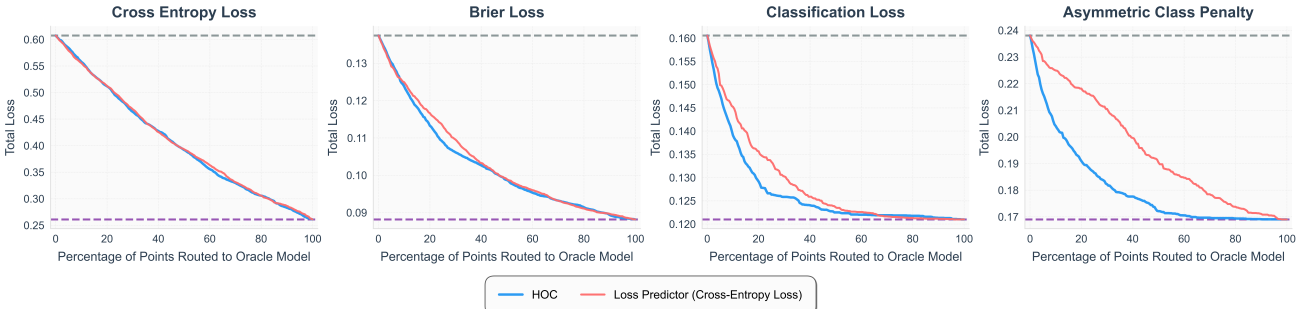
(b) Uncalibrated Weak Model

Figure 13. Routing performances for different target loss functions for the CIFAR-10H dataset. The loss predictor baseline is a static loss predictor trained to predict the amount of reducible cross-entropy loss. The right (b) employs an out-of-the-box weak predictor for the dataset, whereas the left (a) uses the same predictor but with an additional post-hoc calibration step on the same buckets used by the higher-order-calibrated router.

H.4. SNLI



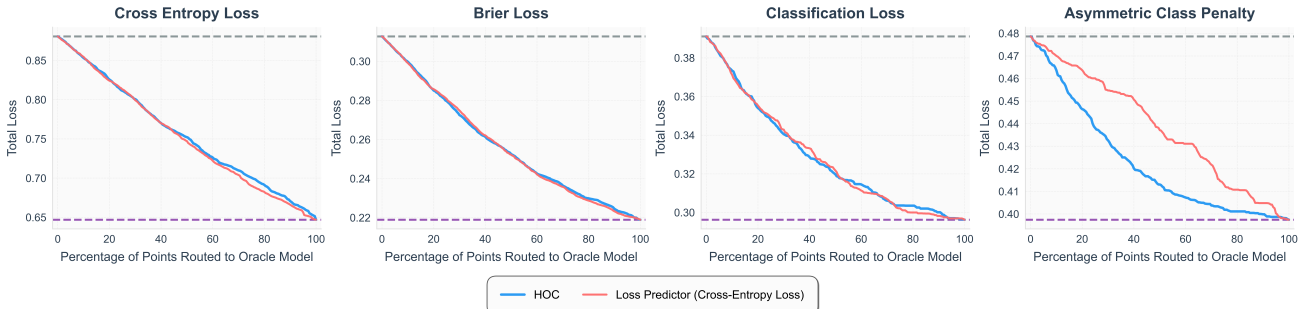
(a) Calibrated Weak Model



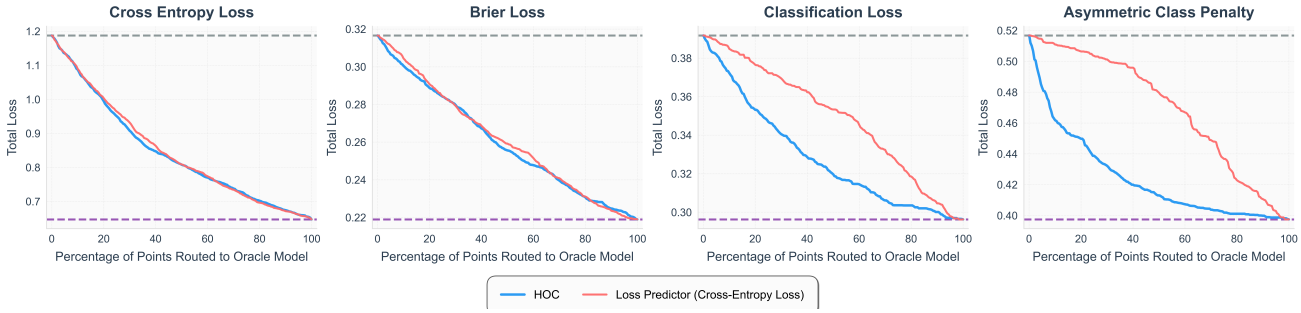
(b) Uncalibrated Weak Model

Figure 14. Routing performances for different target loss functions for the SNLI dataset. The loss predictor baseline is a static loss predictor trained to predict the amount of reducible cross-entropy loss. The right (b) employs an out-of-the-box weak predictor for the dataset, whereas the left (a) uses the same predictor but with an additional post-hoc calibration step on the same buckets used by the higher-order-calibrated router.

H.5. ChaosNLI



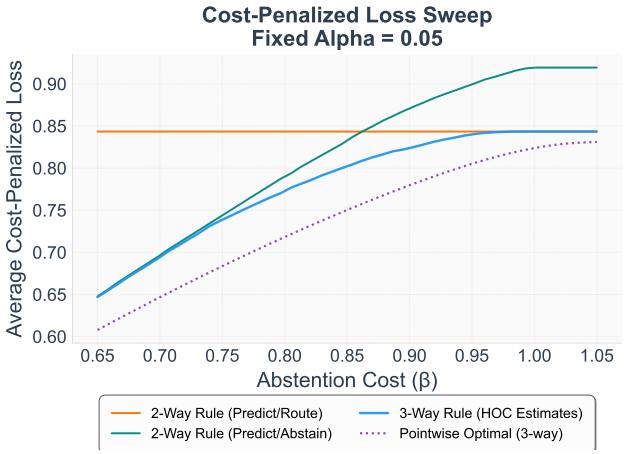
(a) Calibrated Weak Model



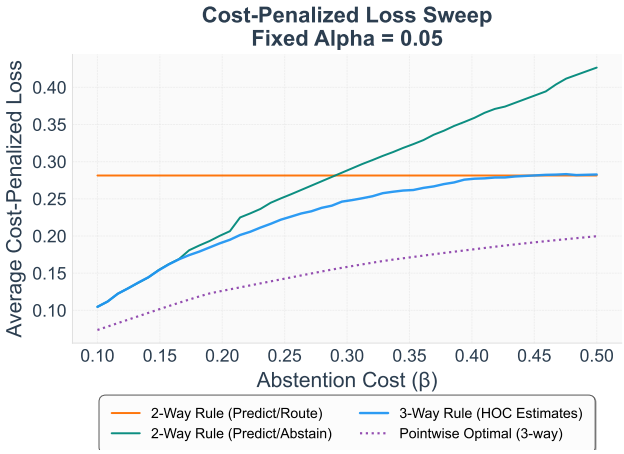
(b) Uncalibrated Weak Model

Figure 15. Routing performances for different target loss functions for the ChaosNLI dataset. The loss predictor baseline is a static loss predictor trained to predict the amount of reducible cross-entropy loss. The right (b) employs an out-of-the-box weak predictor for the dataset, whereas the left (a) uses the same predictor but with an additional post-hoc calibration step on the same buckets used by the higher-order-calibrated router.

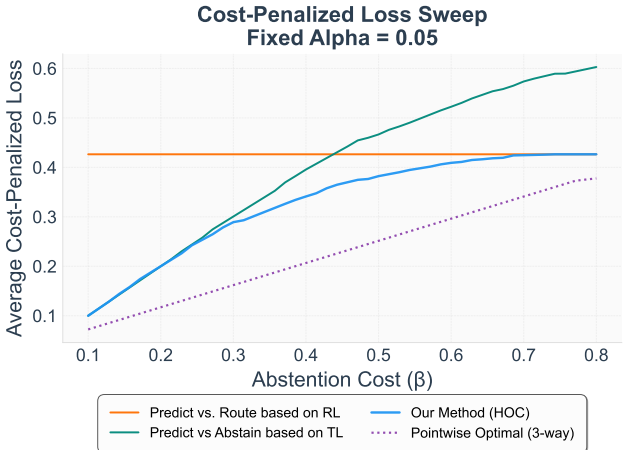
I. Plots of Three-Way Decisions



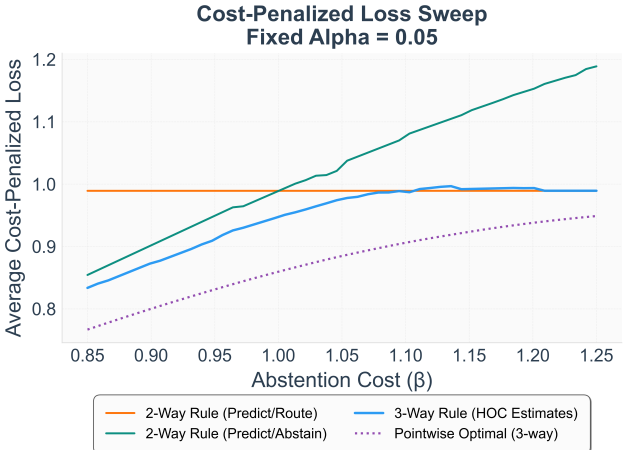
(a) Synthetic Data (target function from Figure 5a)



(b) CIFAR-10H



(c) SNLI



(d) ChaosNLI

Figure 16. Depiction of our method's cost-penalized performance on the three-way decision task of Predict vs Route vs Abstain across four datasets, for a fixed routing penalty of $\alpha = 0.05$ while sweeping the abstention penalty β. We see that our method gracefully adapts to the three-way decision setting, out-performing or closely aligning with the two-way decision baselines of predict/route (orange) and predict/abstain (green).