DYNAMIC LLM DECODING LENGTH PREDICTION: LEARNING FROM BRANCHING FUTURES

Anonymous authors
Paper under double-blind review

ABSTRACT

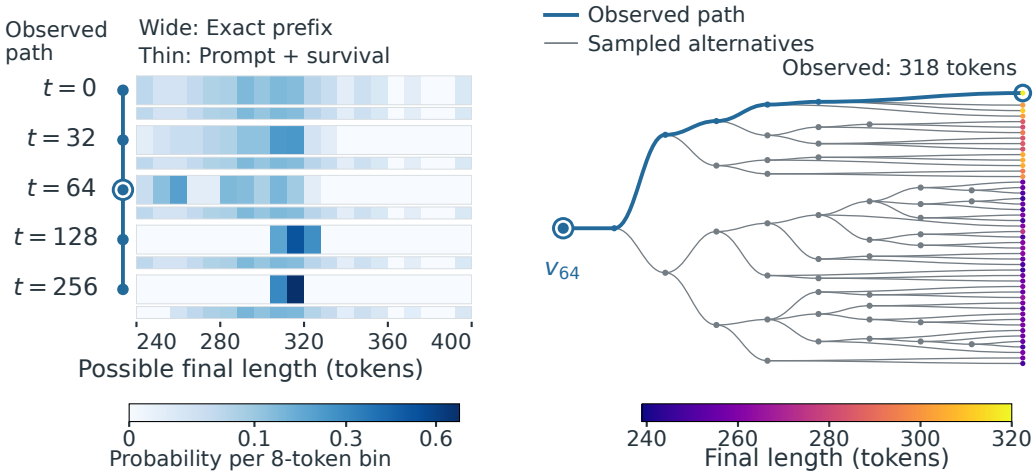
Length prediction during large language model (LLM) decoding provides an important signal for allocating limited computational resources across concurrent requests. Existing decode-time predictors typically learn from sampled trajectories, each of which reveals only one realized future at every visited prefix. However, a prefix admits multiple possible futures, and its remaining-length distribution evolves as more response content is observed. The challenge is therefore to obtain informative conditional supervision without exhaustively exploring this branching future space. In this paper, we propose *BRANching-supervised Distribution prediction* (BranD), an approach for learning prefix-conditioned remaining-length distributions under a limited continuation budget. BranD combines current and historical target-model representations to update its predictions as decoding proceeds. For training, it samples independent continuations from selected prefixes to refine conditional distribution targets, while reusing their paths to expand prefix coverage. An adaptive allocation strategy combines estimated conditional variability with predictor sensitivity to determine how many continuations to sample at each prefix. We provide a fixed-feature risk analysis that connects conditional sampling error to shared prediction error and justifies the allocation strategy. Experiments across multiple models and tasks validate the effectiveness of our proposed BranD.

1 INTRODUCTION

Large language models (LLMs) increasingly produce long and heterogeneous generations, from conversational assistance (Ouyang et al., 2022) and mathematical reasoning (Guo et al., 2025) to agentic workflows (Xi et al., 2025). As generation lengths vary substantially across requests, each request's computational demand becomes difficult to anticipate. This is particularly important when many requests compete for time-shared computational resources, where inaccurate estimates of future workload can lead to inefficient memory use, head-of-line blocking, and poor resource utilization, degrading serving efficiency. This concern is central to Computational Resource Efficient Learning (CoRE-Learning) (Zhou, 2024), which studies combining learning and resource allocation. Output-length prediction provides an important signal for allocating these resources across concurrent generation requests. In reinforcement learning post-training, length predictions can guide rollout dispatching and scheduling to reduce delays caused by long-running samples (Zhong et al., 2025). In LLM serving, length prediction supports memory-aware batching and length-aware scheduling, improving GPU utilization and throughput while reducing response latency (Jin et al., 2023; Fu et al., 2024). As LLM workloads become longer and more heterogeneous, accurately estimating how much generation remains becomes increasingly important for efficient resource allocation.

Considerable progress has been made in predicting output length before and during generation. For prediction before decoding, S^3 (Jin et al., 2023) trains an auxiliary model to predict output-length buckets for memory-aware batching, while LTR (Fu et al., 2024) learns the relative ordering of output lengths to guide request scheduling. More recently, ProD (Wang et al., 2026) and TIE (Zheng et al., 2026) concurrently report that the output lengths obtained from repeated generations of a single prompt are heavy-tailed, and both explicitly account for this prompt-conditioned uncertainty. Methodologically, ProD adopts a non-parametric estimate of the prompt-conditioned length distribution, whereas TIE fits a parametric distributional model. Further advances explore length prediction during decoding. TRAIL (Shahout et al., 2025) uses the target LLM's intermediate embeddings to predict length-interval probabilities and combines successive predictions through temporal smoothing. Pro-

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.



(a) Distributions along one observed path.

(b) One path through branching futures.

Figure 1: Sampled futures of Qwen2.5-7B-Instruct on GSM8K. Figure (a): the final-length distribution along one observed path, where each wide strip is the empirical distribution from 128 exact-prefix continuations and its narrow companion is the root outcome conditioned on survival (square-root color mapping, probabilities on the colorbar). Figure (b): the branching structure behind such a path, expanding the circled $\mathbf{v}_{64}$ to show 48 of 154 distinct cached paths from 160 draws (all 29 prefixes at $t = 96$), with the original response highlighted and endpoint color encoding final length. Unbranched chains are compressed; horizontal distance is not meaningful.

gressive Length Prediction (PLP) (Xie et al., 2026) combines entropy-guided prompt representations with generated-token information to update remaining-length estimates. OUTLETS (Wen et al., 2026) further establishes a connection between speculative decoding and length prediction, using a speculative-decoding backbone to obtain trajectory-aware representations of the evolving generation process. Together, these studies establish a strong foundation for predicting generation workloads from both the input prompt and the response revealed during execution.

Despite these advances, how to obtain informative supervision for decode-time length distributions has received less attention. Existing decode-time predictors typically derive remaining-length targets from responses sampled from the target model, with the label at each decoding step determined by the realized trajectory (Shahout et al., 2025; Wen et al., 2026). However, *a prefix does not determine a single future*. As illustrated in Figure 1(a), continuations from the same prefix can have different lengths, and their distribution changes as more response content is observed. These changes concern both the location and concentration of possible final lengths, beyond the effect of elapsed decoding alone. Furthermore, the possible futures form a branching structure. Figure 1(b) shows that a shared prefix can extend into different descendants, each of which still admits continuations of different lengths. A sampled trajectory follows only one branch and supplies one realized future at each visited prefix. Sampling additional continuations can reveal alternative futures and improve the available conditional supervision. However, exhaustively exploring all branches is computationally infeasible. The key challenge is therefore to decide *where to branch and how many continuations to sample*, so as to learn evolving prefix-conditioned length distributions under a limited continuation budget.

To this end, we propose *BRANching-supervised Distribution prediction* (BranD), a framework for learning prefix-conditioned remaining-length distributions under a limited sampling budget. BranD consists of three components: (i) *Dynamic distribution prediction*, which combines current and historical representations of the target model to update the remaining-length distribution as generation proceeds; (ii) *Branching-future-based supervision*, which samples multiple independent continuations from selected prefixes to construct conditional distribution targets, while reusing the generated paths to supervise additional decoding states; and (iii) *Adaptive continuation allocation*, which uses pilot estimates of conditional variability and predictor sensitivity to decide how many continuations to sample at each prefix according to the estimated prediction benefit of further sampling. We provide a fixed-feature risk analysis to characterize how continuation allocation affects shared prediction error and support the allocation rule. Branching is restricted to training; at deployment, the predictor uses only observed generation states and history, with fixed parameters.

2 PROBLEM FORMULATION

In this section, we formulate length prediction as estimating a prefix-conditioned length distribution during autoregressive decoding with fixed target model and decoding configuration.

Remaining-Length Distribution Prediction. For a prompt $\mathbf{x} \in \mathcal{X}$, let $L \leq C$ denote the response length, where an emitted EOS token is counted in L, and reaching C is treated as termination. Let $\mathbf{v}_t = (\mathbf{x}, \mathbf{y}_{1:t})$ denote an active prefix after t response tokens have been generated. The remaining length is $R_t = L - t \in \{1, \ldots, C - t\}$. Under stochastic decoding, the same prefix can yield different continuations, and hence different remaining lengths. We therefore model R_t as a prefix-conditioned random variable with probability mass function $p^\star(\cdot \mid \mathbf{v}_t)$ and cumulative distribution function (CDF)

$$F_{\mathbf{v}_t}(r) = \mathbb{P}(R_t \leq r \mid \mathbf{v}_t) = \sum_{s=1}^{r} p^\star(s \mid \mathbf{v}_t). \tag{1}$$

At $t = 0$, we have $\mathbf{v}_0 = \mathbf{x}$ and $R_0 = L$, so the problem reduces to predicting the prompt-conditioned output-length distribution (Wang et al., 2026). As decoding proceeds, the prefix $\mathbf{v}_t$ grows with the generated tokens, and the target distribution $p^\star(\cdot \mid \mathbf{v}_t)$ is updated accordingly.

Learning from Branching Futures. The learner starts from a fixed prompt set $\mathcal{X}$ and a set D_0 of completed trajectories sampled from these prompts. Each trajectory yields one realized remaining length at every prefix it visits, so a single trajectory reveals only one future per prefix. To obtain more futures, the learner is given a budget of B extra generations. Let $\mathcal{V}$ be the finite set of eligible origins, namely the prompt roots and the prefixes already observed. In round $b \in \{1, \ldots, B\}$, it picks an origin $\mathbf{v}_b \in \mathcal{V}$ and samples one fresh continuation from $\mathbf{v}_b$ under the same decoding configuration and length cap. Each continuation adds a new remaining-length observation at $\mathbf{v}_b$ and at every later prefix it passes through. The choice of $\mathbf{v}_b$ may depend on all outcomes seen in earlier rounds.

Using D_0 together with the B sampled continuations, the learner fits a single predictor $q_\psi(\cdot \mid \mathbf{v})$, with induced CDF $\widehat{F}_\psi(\cdot \mid \mathbf{v})$, shared across all prefixes. Given an evaluation distribution Q that weights prefixes by how much their accuracy matters, the prediction risk is

$$\mathcal{E}_Q(\psi) = \mathbb{E}_{\mathbf{v} \sim Q} \sum_{r=0}^{C-1} \left(\widehat{F}_\psi(r \mid \mathbf{v}) - F_{\mathbf{v}}(r) \right)^2. \tag{2}$$

We measure error on the CDF because remaining length is ordered: mispredicting by many tokens should cost more than mispredicting by a few, which pointwise divergences such as cross-entropy or KL ignore. Since Eq. (2) is exactly the excess expected CRPS, its CDF form yields the finite-sampling decomposition behind our continuation-allocation analysis. Extra continuations are used only for training; at deployment the predictor runs on the observed prefix and target-model states alone, with fixed parameters and no further sampling.

Key Challenge. The difficulty is that estimating each $F_{\mathbf{v}}$ and spending the budget B cannot be separated. Because a trajectory reveals only one future per prefix, learning the variability of $F_{\mathbf{v}}$ requires actively sampling repeated continuations from the same prefix. Yet the prefix space branches rapidly and B is limited, so the learner must decide where to sample and how many futures to draw, all before knowing how much each prefix actually varies. Parameter sharing tightens the coupling: a future sampled at one prefix reshapes predictions everywhere through the shared predictor, so the value of a continuation depends not only on local variability but on its effect on the overall risk.

Existing methods sidestep this trade-off. Prompt-level methods such as ProD model output length before decoding and ignore how $F_{\mathbf{v}}$ evolves with the response. Decode-time predictors such as TRAIL, PLP, and OUTLETS refine estimates from evolving model states, but supervise on ordinary trajectories that expose one future per prefix. Neither addresses how to allocate a limited budget for repeated conditional supervision over a branching future space.

3 Our Approach

In this section, we develop BranD in three steps. We first build a history-aware predictor that updates the remaining-length distribution from the evolving prefix as decoding proceeds (Section 3.1). We then improve its supervision with branching futures, which refine the conditional targets and expand prefix coverage (Section 3.2). Building on this predictor, we derive a predictor-aware allocation rule that decides how many continuations to sample at each candidate prefix (Section 3.3). Finally, we provide the theoretical justification behind these components (Section 3.4).

3.1 Dynamic Distribution Prediction

As decoding proceeds, we predict from the evolving prefix rather than a static initial estimate. Following OUTLETS (Wen et al., 2026), we use a speculative-decoding-style causal backbone over multi-layer target states and observed tokens, augmented with residual access to target-state history and a prefix-conditioned remaining-length distribution head.

For a prompt of length P, let $\boldsymbol{h}_i^{(\ell)}$ and $\mathbf{e}_i$ be the target hidden state at layer ℓ and the token embedding at position i. We project states from a layer set $\mathcal{I}$ and aggregate them with the observed tokens:

$$\mathbf{f}_i = W_f \,\mathrm{Concat}_{\ell\in\mathcal{I}}\, \boldsymbol{h}_i^{(\ell)}, \qquad \mathbf{z}_t = B_\psi\big((\mathbf{f}_{i-1}, \mathbf{e}_i)_{i=2}^{P+t}\big). \tag{3}$$

Here, B_ψ is a causal auxiliary decoder. Each shifted pair combines the preceding target context with the token actually observed. The resulting $\mathbf{z}_t$ summarizes the prompt and response prefix. We further use $\mathbf{z}_t$ as the query in a residual attention module F_ψ, with causally available projected target states as keys and values. The residual attention therefore exposes the readout directly to earlier target states.

A nonlinear head produces logits $\boldsymbol{a}_t = H_\psi(F_\psi(\mathbf{z}_t)) \in \mathbb{R}^C$. For an active prefix $\mathbf{v}_t$, only lengths $1, \ldots, C-t$ are feasible, so we predict

$$q_\psi(r \mid \mathbf{v}_t) = \frac{\exp(a_{t,r})}{\sum_{u=1}^{C-t}\exp(a_{t,u})}, \qquad 1 \le r \le C-t. \tag{4}$$

The probability outside this support is zero. The cumulative sum gives $\widehat{F}_\psi(\cdot \mid \mathbf{v}_t)$, and the median provides a scalar estimate. Thus, the same readout represents both typical remaining lengths and the variation among possible futures. For terminated prefixes, the distribution is a point mass at zero.

Only the auxiliary prediction modules are trained; the target LLM and its token embeddings remain frozen. At inference, prompt prefill initializes the caches, which are updated with each observed response token. Predictions use fixed parameters and require no additional continuation sampling.

3.2 Supervision from Branching Futures

An ordinary trajectory covers multiple prefixes but reveals only one future at each. Starting from $\mathcal{D}_0$, we sample continuations from selected roots and internal prefixes under the same decoding policy. These branching futures improve supervision at two levels: they refine conditional targets at their origins and expand prefix coverage along their paths.

Refining Conditional Targets. For an origin $\mathbf{v}_t$, a continuation with total response length L contributes the remaining length $L-t$. Let $\mathcal{R}_\mathbf{v}$ be the multiset of observations collected at $\mathbf{v}$ from ordinary data and additional continuations, and let $n_\mathbf{v} = |\mathcal{R}_\mathbf{v}|$. They define the empirical target CDF

$$\bar{F}_\mathbf{v}(r) = \frac{1}{n_\mathbf{v}} \sum_{R\in\mathcal{R}_\mathbf{v}} \mathbb{1}_{\{R\le r\}}. \tag{5}$$

Each new outcome refines the target for the same observed condition. Unlike averaging lengths into a scalar label, this construction retains the variation among possible futures.

Expanding Prefix Coverage. The same continuation also supplies $(\mathbf{v}_s, L-s)$ at retained active prefixes strictly after its origin. Observations are pooled when their prompt and response-prefix tokens match exactly; newly visited prefixes become additional training examples. Refinement therefore improves the target at an existing prefix, whereas coverage adds conditions on which the predictor is trained. Both use the same sampled continuations.

Training. Collecting more futures should refine a prefix's target without inflating its training weight. For an outcome multiset $\mathcal{R}_{\mathbf{v}}$, we average the distributional loss within the prefix:

$$\ell_\psi(\mathbf{v}) = \frac{1}{n_{\mathbf{v}}} \sum_{R \in \mathcal{R}_{\mathbf{v}}} \big[\alpha \ \mathrm{CRPS}(\widehat{F}_\psi(\cdot \mid \mathbf{v}), R) - \beta \log q_\psi(R \mid \mathbf{v})\big], \tag{6}$$

where $\alpha, \beta > 0$. CRPS accounts for the ordering of lengths, while the log-likelihood term assigns probability to observed outcomes. For the CRPS term, averaging over outcomes is equivalent, up to a constant independent of ψ, to matching the empirical CDF $\bar{F}_{\mathbf{v}}$. Thus, repeated futures provide a distributional target, not additional distinct conditions.

We apply this loss within three supervision streams: ordinary prefixes, branching origins, and retained path prefixes. Their prompt–prefix-weighted averages are denoted by $\mathcal{L}_D$, $\mathcal{L}_R$, and $\mathcal{L}_P$, respectively; $\mathcal{L}_D$ also retains an auxiliary scalar-length objective. We optimize

$$\mathcal{L}(\psi) = \lambda_D \mathcal{L}_D(\psi) + \lambda_R \mathcal{L}_R(\psi) + \lambda_P \mathcal{L}_P(\psi), \tag{7}$$

with nonnegative stream weights. Separate weighting prevents path states from overwhelming the refined conditional targets. Starting from the ordinary-data predictor, refitting first emphasizes ordinary and origin supervision, then incorporates path supervision while retaining all $\mathcal{D}_0$.

3.3 Predictor-Aware Continuation Allocation

Additional sampling helps when a conditional target is noisy and strongly affects shared predictions. Variability alone misses the second factor: two equally variable prefixes can affect the shared predictor differently. We therefore combine target dispersion with predictor sensitivity.

Estimating Sampling Value. We fix a finite nonempty candidate pool $\mathcal{V} \subseteq \mathcal{V}_0$ containing roots and predetermined internal prefixes, where $\mathcal{V}_0$ denotes the ordinary-data prefixes. Planning features $\mathbf{z}_{\mathbf{u}} \in \mathbb{R}^d$ are extracted from the ordinary-data predictor and frozen before acquisition. Given fixed prefix weights $w_{\mathbf{v}} \geq 0$, a planning distribution Q_{tr} over $\mathcal{V}_0$, and regularization $\gamma > 0$, define

$$V = \gamma I_d + \sum_{\mathbf{v} \in \mathcal{V}} w_{\mathbf{v}} \mathbf{z}_{\mathbf{v}} \mathbf{z}_{\mathbf{v}}^\top, \qquad \Sigma_{\mathrm{tr}} = \mathbb{E}_{\mathbf{u} \sim Q_{\mathrm{tr}}}[\mathbf{z}_{\mathbf{u}} \mathbf{z}_{\mathbf{u}}^\top], \qquad s_{\mathbf{v}} = w_{\mathbf{v}}^2 \mathbf{z}_{\mathbf{v}}^\top V^{-1} \Sigma_{\mathrm{tr}} V^{-1} \mathbf{z}_{\mathbf{v}}. \tag{8}$$

The sensitivity $s_{\mathbf{v}}$ measures how target errors at $\mathbf{v}$ propagate to predictions over Q_{tr}. For independent $R_{\mathbf{v}}, R'_{\mathbf{v}} \sim F_{\mathbf{v}}$, let $A_{\mathbf{v}} = \frac{1}{2}\mathbb{E}\,|R_{\mathbf{v}} - R'_{\mathbf{v}}|$ be the conditional dispersion. Theorem 1 shows that, in a fixed-feature model, $m_{\mathbf{v}}$ independent futures contribute $s_{\mathbf{v}} A_{\mathbf{v}} / m_{\mathbf{v}}$ to the sampling-dependent error.

For each candidate, we first draw two independent pilot continuations and estimate

$$\widehat{A}_{\mathbf{v}} = \frac{1}{2}\left|R^{\mathrm{pil}}_{\mathbf{v},1} - R^{\mathrm{pil}}_{\mathbf{v},2}\right|, \qquad \widehat{a}_{\mathbf{v}} = s_{\mathbf{v}} \widehat{A}_{\mathbf{v}}. \tag{9}$$

The pilot is used only for planning. We then collect fresh production futures for fitting, so that these labels remain independent given the chosen allocation.

Allocating Continuations. We measure the integer budget B in continuation calls. After the pilot, the production budget is $B_{\mathrm{prod}} = B - 2\,|\mathcal{V}|$. For $B_{\mathrm{prod}} \geq |\mathcal{V}|$ and an integer cap $U \geq 1$, we choose integer counts by solving

$$\begin{aligned} \min_{\boldsymbol{m}} \quad & \sum_{\mathbf{v} \in \mathcal{V}} \frac{\widehat{a}_{\mathbf{v}}}{m_{\mathbf{v}}} \\ \text{subject to} \quad & 1 \leq m_{\mathbf{v}} \leq U, \qquad \sum_{\mathbf{v} \in \mathcal{V}} m_{\mathbf{v}} \leq B_{\mathrm{prod}}. \end{aligned} \tag{10}$$

Write $\mathcal{M}_B$ for its feasible set. Starting from one future per origin, we repeatedly assign an extra continuation to a feasible origin with the largest marginal gain

$$\Delta_{\mathbf{v}}(\boldsymbol{m}) = \frac{\widehat{a}_{\mathbf{v}}}{m_{\mathbf{v}}(m_{\mathbf{v}} + 1)}. \tag{11}$$

We stop when the budget is exhausted or no positive feasible gain remains. As more futures are assigned to an origin, its gain decreases, directing later calls toward other origins where refinement has greater estimated value. Under unit call costs, this rule exactly solves Eq. (10), as established below. The counts are committed before production; the resulting futures provide both origin and path supervision in Eq. (7). Implementation details are given in Appendix C.2.

3.4 Theoretical Justification

We analyze fixed-feature CDF regression to isolate the effect of conditional supervision. Condition on the ordinary data and shared history, with the planning quantities in Eq. (8) fixed before the pilot. Assume $\mathbb{E}_{\mathbf{u}\sim Q_{\mathrm{tr}}}\|\mathbf{z}_{\mathbf{u}}\|_2^2 < \infty$. Let $\bar{F}_{\mathbf{v}}^{\mathrm{prod}}$ be the empirical CDF of $m_{\mathbf{v}}$ fresh direct futures, represented on $r = 0, \ldots, C-1$. Consider

$$\widehat{\Theta}(\boldsymbol{m}) = \operatorname*{arg\,min}_{\Theta\in\mathbb{R}^{d\times C}} \sum_{\mathbf{v}\in\mathcal{V}} w_{\mathbf{v}} \left\|\Theta^\top \mathbf{z}_{\mathbf{v}} - \bar{F}_{\mathbf{v}}^{\mathrm{prod}}\right\|_2^2 + \gamma \|\Theta\|_{\mathrm{F}}^2 . \tag{12}$$

The oracle-label learner Θ° uses the same fitting rule with true conditional CDFs $F_{\mathbf{v}}$ in place of the empirical targets. Write $\mathcal{E}_{Q_{\mathrm{tr}}}(\Theta) = \mathbb{E}_{\mathbf{u}\sim Q_{\mathrm{tr}}} \left\|\Theta^\top \mathbf{z}_{\mathbf{u}} - F_{\mathbf{u}}\right\|_2^2$.

Theorem 1 (Prediction error)**.** *Conditional on the pilot, suppose that the production counts are fixed and the futures are independent within and across origins, with CDF $F_{\mathbf{v}}$ at origin $\mathbf{v}$. Then*

$$\mathbb{E}_{\mathrm{prod}}\mathcal{E}_{Q_{\mathrm{tr}}}(\widehat{\Theta}(\boldsymbol{m})) = \mathcal{E}_{Q_{\mathrm{tr}}}(\Theta^\circ) + \sum_{\mathbf{v}\in\mathcal{V}} \frac{s_{\mathbf{v}} A_{\mathbf{v}}}{m_{\mathbf{v}}}. \tag{13}$$

The first term is the error with exact conditional labels, and the second is the additional error from finite sampling. No linear realizability assumption is required. An extra future at $\mathbf{v}$ reduces the latter by $s_{\mathbf{v}}A_{\mathbf{v}}/[m_{\mathbf{v}}(m_{\mathbf{v}}+1)]$, yielding Eq. (11) after pilot estimation. The following lemma characterizes this allocation and its sensitivity to pilot error.

Lemma 1 (Continuation allocation)**.** *Let $J(\boldsymbol{m}) = \sum_{\mathbf{v}\in\mathcal{V}} s_{\mathbf{v}}A_{\mathbf{v}}/m_{\mathbf{v}}$ and $\widehat{J}(\boldsymbol{m}) = \sum_{\mathbf{v}\in\mathcal{V}} s_{\mathbf{v}}\widehat{A}_{\mathbf{v}}/m_{\mathbf{v}}$. The marginal rule returns a minimizer $\widehat{\boldsymbol{m}}$ of $\widehat{J}$ over $\mathcal{M}_B$. For any minimizer $\boldsymbol{m}^\star$ of J there,*

$$0 \le J(\widehat{\boldsymbol{m}}) - J(\boldsymbol{m}^\star) \le \sum_{\mathbf{v}\in\mathcal{V}} s_{\mathbf{v}} \left|\widehat{A}_{\mathbf{v}} - A_{\mathbf{v}}\right|. \tag{14}$$

The lemma separates exact optimization of the estimated objective from error in estimating sampling value. Both results concern independent direct futures and fixed planning features, not nonlinear refitting or path-coverage gains. The detailed proofs are given in Appendix B.

4 Experiments

In this section, we evaluate BranD for remaining-length prediction across six model–task settings, ablate each component, and apply it to LLM serving.

4.1 Experimental Setup

Models and datasets. We use Qwen2.5-7B-Instruct (Qwen et al., 2024) and Llama-3-8B-Instruct as target models. The experiments cover mathematical reasoning on GSM8K (Cobbe et al., 2021), code generation on MBPP (Austin et al., 2021), and conversation on first-turn LMSYS-Chat-1M requests (Zheng et al., 2024), denoted by Math, Coding, and Chat, respectively. Predictors are trained separately for each model–task pair, with the target model and decoding configuration fixed within each setting. The response-length cap is $C = 2048$, except for Qwen Chat, where $C = 4096$.

Baselines. We compare BranD with TRAIL (Shahout et al., 2025), PLP (Xie et al., 2026), OUTLETS (Wen et al., 2026), and ProD (Wang et al., 2026), which uses survival conditioning to update its prompt-level length distribution during decoding. We include Empirical as a featureless baseline: it conditions the training-length distribution on survival to the current decoding stage. All baselines use the complete ordinary trajectory collection D_0.

Evaluation. We report mean absolute error (MAE) for point prediction and the continuous ranked probability score (CRPS) for distribution prediction. Both metrics are in tokens; lower is better. For MAE, ProD and BranD use the median of its predictive distribution. For CRPS, scalar outputs are evaluated as point masses. We evaluate active prefixes after 32, 64, 128, and 256 generated tokens, using independently sampled continuations from each prefix. All methods share the same evaluation prefixes and continuations. Scores are averaged over continuations within each prefix, then over prefixes within each prompt, and finally over prompts. Each method uses a single fixed model across all decoding stages. Implementation, training, data, and generation details are in Appendix C.

Table 1: Remaining-length prediction errors in tokens (lower is better). The best result in each column is bold. Avg. denotes the mean of each metric across the three tasks.

	Math		Coding		Chat		Avg.	
Method	MAE	CRPS	MAE	CRPS	MAE	CRPS	MAE	CRPS
Qwen2.5-7B-Instruct								
Empirical	62.654	44.402	98.204	69.938	291.729	212.074	150.862	108.805
ProD	46.429	33.763	79.043	65.070	149.551	113.262	91.674	70.698
TRAIL	52.923	40.738	62.424	46.803	177.010	129.835	97.452	72.459
PLP	43.695	52.738	51.127	69.888	170.304	157.802	88.375	93.476
OUTLETS	35.371	35.371	44.171	44.171	116.238	116.238	65.260	65.260
BranD	**34.137**	**24.990**	**43.214**	**31.718**	**108.684**	**86.203**	**62.012**	**47.637**
Llama-3-8B-Instruct								
Empirical	37.401	27.185	66.163	46.293	205.821	143.068	103.128	72.182
ProD	31.431	23.131	64.725	47.139	96.279	70.875	64.145	47.048
TRAIL	54.129	36.181	57.922	41.596	98.906	74.151	70.319	50.643
PLP	24.629	28.863	35.928	35.486	106.563	105.506	55.707	56.618
OUTLETS	18.672	18.672	33.054	33.054	70.731	70.731	40.819	40.819
BranD	**18.123**	**13.475**	**31.625**	**23.060**	**67.461**	**50.910**	**39.070**	**29.148**

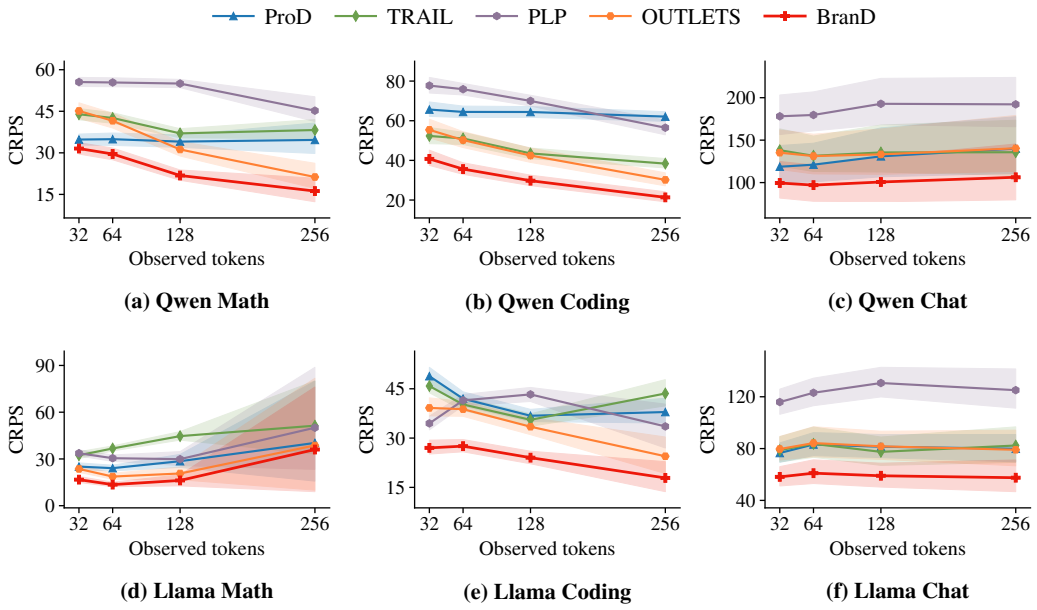


Figure 2: Remaining-length CRPS at different decoding stages. Shaded bands show descriptive pointwise 95% prompt-bootstrap intervals (1,000 resamples).

4.2 Remaining-Length Prediction

Overall prediction accuracy. Table 1 summarizes the results. BranD achieves the lowest MAE and CRPS in all six settings. Its average MAE is 62.012 on Qwen and 39.070 on Llama, improving over OUTLETS, the strongest baseline in point prediction, by 5.0% and 4.3%, respectively. The corresponding average CRPS reductions are 27.0% and 28.6%. The advantage also holds against probabilistic predictors: on Qwen Math, BranD reduces CRPS from 33.763 for ProD with survival conditioning to 24.990, an improvement of 26.0%. These results show BranD improves distribution prediction while providing more accurate point estimates across models and tasks.

Prediction throughout decoding. Figure 2 compares CRPS at different decoding stages. BranD achieves the lowest CRPS at every evaluated stage across the six settings. On Qwen Math, its

Table 2: Component ablation of BranD. MAE (left) and CRPS (right) in tokens under the shared decode-time protocol. Lower is better; best in each column is bold. Scalar predictions are scored as point masses for CRPS.

Components		MAE						CRPS					
		Qwen			Llama			Qwen			Llama		
Predictor	Branch	Math	Coding	Chat	Math	Coding	Chat	Math	Coding	Chat	Math	Coding	Chat
–	–	35.37	44.17	116.24	18.67	33.05	70.73	35.37	44.17	116.24	18.67	33.05	70.73
✓	–	34.89	44.18	108.78	18.13	31.67	67.51	26.60	32.88	86.38	13.50	23.12	51.14
–	✓	34.91	43.57	**107.58**	18.61	32.52	68.68	34.91	43.57	107.58	18.61	32.52	68.68
✓	✓	**34.14**	**43.21**	108.68	**18.12**	**31.63**	**67.46**	**24.99**	**31.72**	**86.20**	**13.48**	**23.06**	**50.91**

improvement over OUTLETS ranges from 23.8% to 30.1% across the four stages. The advantage also persists in both Chat settings, where the error curves are comparatively flat. Thus, BranD's overall improvement is maintained across prefixes of different lengths, rather than being concentrated at a particular decoding stage.

4.3 Ablation Study

We ablate the two components that BranD adds on top of the scalar base predictor: the distribution predictor and branching supervision. We compare four variants: the scalar base, the base with branching supervision, the distribution predictor without branching, and the full BranD combining both. All variants share the ordinary trajectory collection D_0, the same splits and labels, and matched training configurations, differing only in these two components, and are scored under the main experiments' decode-time protocol. Setting-specific configurations are provided in Appendix C.

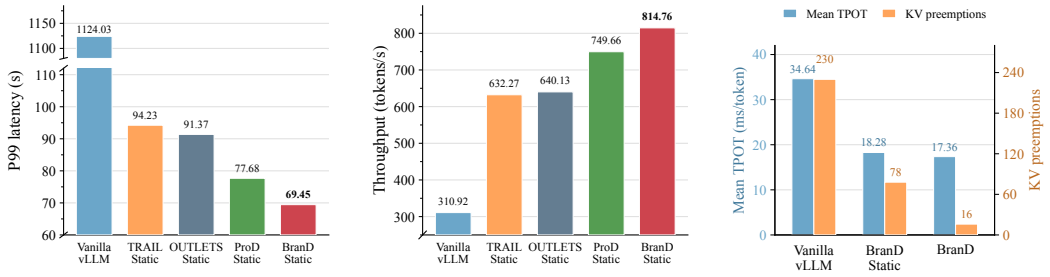
Table 2 reports the results. First, the distribution predictor lowers CRPS in all six settings, since it replaces point masses with calibrated distributions; on Qwen Math CRPS drops from 36.05 to 26.60. Second, branching supervision lowers MAE in every setting for the scalar base, so its benefit is not tied to the distributional head. Finally, combining the two yields the full BranD, which attains the lowest MAE in five of six settings and the lowest CRPS in all six; on Qwen Math it reaches 34.14 MAE and 24.99 CRPS. The only exception is Qwen Chat MAE, where branching alone achieves 107.58 compared with 108.68 for full BranD. Overall, the results support combining distribution prediction with branching supervision.

4.4 Experiments on LLM Serving

To test whether length prediction can benefit LLM serving, we consider two scenarios: (i) admission and ordering control for incoming requests, and (ii) continuous KV budgeting when requests arrive in waves. To create a high-pressure regime, we tighten both the KV-cache budgets and the concurrency: both settings run Qwen2.5-7B-Instruct on LMSYS-Chat in a disaggregated vLLM (Kwon et al., 2023) deployment with one prefill and one decode worker, each on an A100 80GB GPU, capped at 2 and 0.5 GiB of paged KV cache and 64 sequences per worker. Each forecast sets a logical KV admission budget, while vLLM keeps its usual on-demand physical KV allocation.

Admission and ordering control. In this experiment, we consider 192 first-turn LMSYS-Chat requests that all arrive at once, so the full load is present from the start and each predictor must guide admission and ordering. The requests generate freely under a $4{,}096$-token cap, and each forecast also sets an initial decode priority. We compare Vanilla vLLM (FCFS with on-demand KV) against TRAIL, OUTLETS, ProD, and BranD. Since ProD and BranD predict a distribution, we use its median (quantile 50) as the length estimate. All methods share the same prompts, order, and seeds, and we report all-request P99 latency and full-drain throughput.

The results are shown in Figures 3(a) and 3(b). Vanilla vLLM admits requests in arrival order and grows KV on demand, so under the concurrent load it over-commits decode memory and preempts running requests, triggering 756 preemptions and forcing 73 requests to the output cap. TRAIL, OUTLETS, and ProD instead use length predictions to admit and order requests within the KV budget, avoiding collapse and draining the batch. BranD improves further because it estimates remaining length more accurately and sizes KV admission from the median prediction (quantile 50), reducing



(a) P99 end-to-end latency. **(b) Output-token throughput.** **(c) TPOT and KV preemptions.**

Figure 3: Serving performance for Qwen2.5-7B-Instruct on LMSYS-Chat under KV pressure.

preemptions. As a result, it attains the lowest P99 at 69.45 s (24.00% below OUTLETS and 10.60% below ProD) and the highest throughput at 814.76 tokens/s (27.28% and 8.68% above).

Continuous KV budgeting under wave arrivals. In this experiment, we consider 192 requests that arrive in three waves at 0, 10, and 20 s, replaying 64 LMSYS-Chat trajectories per wave. All run real forward passes along the same trajectories, so every method completes the same 82,887 output tokens, and differences come from scheduling, not generation. We compare Vanilla vLLM against BranD-Static and BranD, where BranD-Static uses only the initial prediction and updates it by subtracting the decoded length, and BranD recomputes its remaining-length distribution from the current observed prefix and target-model states during decoding. Both share one controller and budget KV from the median ($Q50$) over a 128-token horizon at a 90% threshold. We report native KV preemptions and mean TPOT (over the 144 requests with ≥ 2 content tokens).

As shown in Figure 3(c), predictive KV budgeting substantially reduces memory pressure during serving. Vanilla vLLM performs no proactive KV reservation and incurs 230 KV preemptions. Static BranD budgeting reduces this number to 78, while survival-conditioned budgeting further reduces it to only 16. Mean TPOT follows the same trend, decreasing from 34.64 ms for Vanilla vLLM to 18.28 ms with static budgeting and 17.36 ms with survival conditioning. This improvement is consistent with fewer KV preemptions, which reduce recomputation overhead during decoding.

5 Conclusion and Future Work

We study decode-time length prediction for LLMs and argue that the target is a *prefix-conditioned remaining-length distribution* that evolves and branches as decoding proceeds, so an ordinary trajectory, which gives one future per prefix, is a thin training signal. The difficulty is to obtain conditional supervision under a limited continuation budget. To this end, we propose BranD, which couples a *dynamic predictor* that reuses the target model's states to update the distribution from the observed prefix; *branching-future-based supervision* that samples independent continuations to sharpen conditional targets and label additional states; and *predictor-aware allocation* that spends the budget where a prefix both varies more and affects shared predictions more. A fixed-feature analysis separates the error into oracle-label and finite-sampling terms and shows our rule minimizes the latter. Branching stays in training; at deployment, forecasts use observed states with fixed parameters and no sampling. Across math, coding, and chat on Qwen-2.5-7B and Llama-3-8B, BranD attains the lowest average MAE and CRPS, cutting distributional error by up to roughly 30% over the strongest baselines, with gains that persist across decoding stages and carry over to serving under KV pressure.

Our study focuses on a fixed target model and decoding policy at serving time. In the future, we aim to extend our approach to evolving models and policies, particularly during RL post-training, broaden the analysis to cover these effects, and integrate forecast updates with uncertainty-aware resource allocation for LLM serving and rollout generation.

AI USE STATEMENT

In this work, we used generative AI tools to assist in the writing of proofs, provide feedback on experiments, and implement methods. We have not used generative AI tools to generate synthetic data sets, help develop theoretical models or conceptual frameworks, formulate mathematical claims, provide critical ingredients for proving mathematical claims, propose or refine hypotheses, or interpret results; the remaining required-disclosure tasks are not applicable to this work. Additionally, we used generative AI tools to suggest experimental parameters, summarize or analyse existing literature, discover research topics or identify gaps, and edit the paper to improve readability. We have reviewed all AI-assisted work. LLM-generated code assistance were verified and tested for correctness by the authors. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

REPRODUCIBILITY STATEMENT

We provide complete proofs of the theoretical results in Appendix B and detailed descriptions of the models, datasets, decoding configurations, continuation acquisition procedure, baseline implementations, and evaluation protocol in Appendix C. Additional experimental results and diagnostics are reported in Appendix D.

REFERENCES

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

Andrew An Bian, Joachim M. Buhmann, Andreas Krause, and Sebastian Tschiatschek. Guarantees for greedy maximization of non-submodular functions with applications. In *Proceedings of the 34th International Conference on Machine Learning (ICML)*, pp. 498–507, 2017.

Alexandra Carpentier and Rémi Munos. Finite time analysis of stratified sampling for Monte Carlo. In *Advances in Neural Information Processing Systems 24 (NIPS)*, pp. 1278–1286, 2011.

Alexandra Carpentier and Rémi Munos. Toward optimal stratification for stratified Monte-Carlo integration. In *Proceedings of the 30th International Conference on Machine Learning (ICML)*, pp. 28–36, 2013.

Ke Cheng, Wen Hu, Zhi Wang, Peng Du, Jianguo Li, and Sheng Zhang. Enabling efficient batch serving for LMaaS via generation length prediction. In *Proceedings of the IEEE International Conference on Web Services (ICWS)*, pp. 853–864, 2024.

Seungbeom Choi, Jeonghoe Goo, Eunjoo Jeon, Mingyu Yang, and Minsung Jang. ELIS: Efficient LLM iterative scheduling system with response length predictor. *ArXiv preprint arXiv:2505.09142*, 2025.

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

Fabian Dickmann and Nikolaus Schweizer. Faster comparison of stopping times by nested conditional Monte Carlo. *The Journal of Computational Finance*, 20(2):101–123, 2016.

Dujian Ding, Ankur Mallick, Shaokun Zhang, Chi Wang, Daniel Madrigal, Mirian Del Carmen Hipolito Garcia, Menglin Xia, Laks V. S. Lakshmanan, Qingyun Wu, and Victor Rühle. BEST-Route: Adaptive LLM routing with test-time optimal compute. In *Proceedings of the 42nd International Conference on Machine Learning (ICML)*, pp. 13870–13884, 2025a.

Yifeng Ding, Hantian Ding, Shiqi Wang, Qing Sun, Varun Kumar, and Zijian Wang. Planning-aware code infilling via horizon-length prediction. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 32930–32942, 2025b.

Roy Eisenstadt, Itamar Zimerman, and Lior Wolf. Overclocking LLM reasoning: Monitoring and controlling thinking path lengths in LLMs. *ArXiv preprint arXiv:2506.07240*, 2025.

Xavier Fontaine, Pierre Perrault, Michal Valko, and Vianney Perchet. Online A-optimal design and active linear regression. In *Proceedings of the 38th International Conference on Machine Learning (ICML)*, pp. 3374–3383, 2021.

Yichao Fu, Siqi Zhu, Runlong Su, Aurick Qiao, Ion Stoica, and Hao Zhang. Efficient LLM scheduling by learning to rank. In *Advances in Neural Information Processing Systems 37 (NeurIPS)*, pp. 59006–59029, 2024.

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Peiyi Wang, Qihao Zhu, Runxin Xu, Ruoyu Zhang, Shirong Ma, Xiao Bi, Xiaokang Zhang, Xingkai Yu, Yu Wu, Z. F. Wu, Zhibin Gou, Zhihong Shao, Zhuoshu Li, Ziyi Gao, Aixin Liu, Bing Xue, Bingxuan Wang, Bochao Wu, Bei Feng, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chong Ruan, Damai Dai, Deli Chen, Dongjie Ji, Erhang Li, Fangyun Lin, Fucong Dai, Fuli Luo, Guangbo Hao, Guanting Chen, Guowei Li, Hao Zhang, Hanwei Xu, Honghui Ding, Huazuo Gao, Hui Qu, Hui Li, Jianzhong Guo, Jiashi Li, Jingchang Chen, Jingyang Yuan, Jinhao Tu, Junjie Qiu, Junlong Li, J. L. Cai, Jiaqi Ni, Jian Liang, Jin Chen, Kai Dong, Kai Hu, Kaichao You, Kaige Gao, Kang Guan, Kexin Huang, Kuai Yu, Lean Wang, Lecong Zhang, Liang Zhao, Litong Wang, Liyue Zhang, Lei Xu, Leyi Xia, Mingchuan Zhang, Minghua Zhang, Minghui Tang, Mingxu Zhou, Meng Li, Miaojun Wang, Mingming Li, Ning Tian, Panpan Huang, Peng Zhang, Qiancheng Wang, Qinyu Chen, Qiushi Du, Ruiqi Ge, Ruisong Zhang, Ruizhe Pan, Runji Wang, R. J. Chen, R. L. Jin, Ruyi Chen, Shanghao Lu, Shangyan Zhou, Shanhuang Chen, Shengfeng Ye, Shiyu Wang, Shuiping Yu, Shunfeng Zhou, Shuting Pan, S. S. Li, Shuang Zhou, Shaoqing Wu, Tao Yun, Tian Pei, Tianyu Sun, Tao Wang, Wangding Zeng, Wen Liu, Wenfeng Liang, Wenjun Gao, Wenqin Yu, Wentao Zhang, W. L. Xiao, Wei An, Xiaodong Liu, Xiaohan Wang, Xiaokang Chen, Xiaotao Nie, Xin Cheng, Xin Liu, Xin Xie, Xingchao Liu, Xinyu Yang, Xinyuan Li, Xuecheng Su, Xuheng Lin, X. Q. Li, Xiangyue Jin, Xiaojin Shen, Xiaosha Chen, Xiaowen Sun, Xiaoxiang Wang, Xinnan Song, Xinyi Zhou, Xianzu Wang, Xinxia Shan, Y. K. Li, Y. Q. Wang, Y. X. Wei, Yang Zhang, Yanhong Xu, Yao Li, Yao Zhao, Yaofeng Sun, Yaohui Wang, Yi Yu, Yichao Zhang, Yifan Shi, Yiliang Xiong, Ying He, Yishi Piao, Yisong Wang, Yixuan Tan, Yiyang Ma, Yiyuan Liu, Yongqiang Guo, Yuan Ou, Yuduan Wang, Yue Gong, Yuheng Zou, Yujia He, Yunfan Xiong, Yuxiang Luo, Yuxiang You, Yuxuan Liu, Yuyang Zhou, Y. X. Zhu, Yanping Huang, Yaohui Li, Yi Zheng, Yuchen Zhu, Yunxian Ma, Ying Tang, Yukun Zha, Yuting Yan, Z. Z. Ren, Zehui Ren, Zhangli Sha, Zhe Fu, Zhean Xu, Zhenda Xie, Zhengyan Zhang, Zhewen Hao, Zhicheng Ma, Zhigang Yan, Zhiyu Wu, Zihui Gu, Zijia Zhu, Zijun Liu, Zilin Li, Ziwei Xie, Ziyang Song, Zizheng Pan, Zhen Huang, Zhipeng Xu, Zhongyu Zhang, and Zhen Zhang. DeepSeek-R1 incentivizes reasoning in LLMs through reinforcement learning. *Nature*, 645(8081):633–638, 2025.

Yunho Jin, Chun-Feng Wu, David Brooks, and Gu-Yeon Wei. S^3: Increasing GPU utilization during generative inference for higher throughput. In *Advances in Neural Information Processing Systems 36 (NeurIPS)*, pp. 18015–18027, 2023.

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

Junyan Li, Wenshuo Zhao, Yang Zhang, and Chuang Gan. Steering LLM thinking with budget guidance. In *Findings of the Association for Computational Linguistics: ACL 2026*, pp. 37437–37451, 2026.

Long-Fei Li, Yu-Yang Qian, Peng Zhao, and Zhi-Hua Zhou. Provably efficient online RLHF with one-pass reward modeling. In *Advances in Neural Information Processing Systems 38 (NeurIPS)*, pp. 166987–167016, 2025.

Hunter Lightman, Vineet Kosaraju, Yuri Burda, Harrison Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. Let's verify step by step. In *Proceedings of the 12th International Conference on Learning Representations (ICLR)*, pp. 39578–39601, 2024.

Rohin Manvi, Joey Hong, Tim Seyde, Maxime Labonne, Mathias Lechner, and Sergey Levine. Zero-overhead introspection for adaptive test-time compute. In *Proceedings of the 14th International Conference on Learning Representations (ICLR)*, pp. 109705–109723, 2026.

Mohamed Amine Merzouk, Dmitri Carpov, Mirko Bronzi, Damiano Fornasiere, and Adam Oberman. How much is left? LLMs linearly encode their remaining output length. *ArXiv preprint arXiv:2607.05316*, 2026.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, John Schulman, Jacob Hilton, Fraser Kelton, Luke Miller, Maddie Simens, Amanda Askell, Peter Welinder, Paul F. Christiano, Jan Leike, and Ryan Lowe. Training language models to follow instructions with human feedback. In *Advances in Neural Information Processing Systems 35 (NeurIPS)*, pp. 27730–27744, 2022.

Grzegorz Piotrowski, Mateusz Bystroński, Mikołaj Hołysz, Jakub Binkowski, Grzegorz Chodak, and Tomasz Jan Kajdanowicz. When will the tokens end? graph-based forecasting for LLMs output length. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Student Research Workshop)*, pp. 843–848, 2025.

Haoran Qiu, Weichao Mao, Archit Patke, Shengkun Cui, Saurabh Jha, Chen Wang, Hubertus Franke, Zbigniew T. Kalbarczyk, Tamer Başar, and Ravishankar K. Iyer. Efficient interactive LLM serving with proxy model-based sequence length prediction. In *Proceedings of the 5th International Workshop on Cloud Intelligence and AIOps (AIOps)*, 2024.

Qwen, An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, et al. Qwen2.5 technical report. *ArXiv preprint arXiv:2412.15115*, 2024.

Amrith Setlur, Chirag Nagpal, Adam Fisch, Xinyang Geng, Jacob Eisenstein, Rishabh Agarwal, Alekh Agarwal, Jonathan Berant, and Aviral Kumar. Rewarding progress: Scaling automated process verifiers for LLM reasoning. In *Proceedings of the 13th International Conference on Learning Representations (ICLR)*, pp. 60808–60838, 2025.

Rana Shahout, Eran Malach, Chunwei Liu, Weifan Jiang, Minlan Yu, and Michael Mitzenmacher. Don't stop me now: Embedding based scheduling for LLMs. In *Proceedings of the 13th International Conference on Learning Representations (ICLR)*, pp. 63345–63368, 2025.

Charlie Snell, Jaehoon Lee, Kelvin Xu, and Aviral Kumar. Scaling LLM test-time compute optimally can be more effective than scaling parameters for reasoning. In *Proceedings of the 13th International Conference on Learning Representations (ICLR)*, pp. 10131–10165, 2025.

Jing Wang, Yu-Yang Qian, Ke Xue, Chao Qian, Peng Zhao, and Zhi-Hua Zhou. Robust length prediction: A perspective from heavy-tailed prompt-conditioned distributions. *ArXiv preprint arXiv:2604.07931*, 2026.

Peiyi Wang, Lei Li, Zhihong Shao, Runxin Xu, Damai Dai, Yifei Li, Deli Chen, Yu Wu, and Zhifang Sui. Math-Shepherd: Verify and reinforce LLMs step-by-step without human annotations. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Long Papers)*, pp. 9426–9439, 2024.

Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc V. Le, Ed H. Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. Self-consistency improves chain of thought reasoning in language models. In *Proceedings of the 11th International Conference on Learning Representations (ICLR)*, 2023.

Weihuang Wen, Yingying Liu, Yichuan Liu, Wenqi Zeng, Li Zhou, Chumin Sun, Jie Sun, and Tianshu Yu. OUTLETS: Output-length prediction from speculative decoding backbones. In *Proceedings of the 2026 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. to appear, 2026.

Zhiheng Xi, Wenxiang Chen, Xin Guo, Wei He, Yiwen Ding, Boyang Hong, Ming Zhang, Junzhe Wang, Senjie Jin, Enyu Zhou, Rui Zheng, Xiaoran Fan, Xiao Wang, Limao Xiong, Yuhao Zhou, Weiran Wang, Changhao Jiang, Yicheng Zou, Xiangyang Liu, Zhangyue Yin, Shihan Dou, Rongxiang Weng, Wenjuan Qin, Yongyan Zheng, Xipeng Qiu, Xuanjing Huang, Qi Zhang, and Tao Gui.

The rise and potential of large language model based agents: A survey. *Science China Information Sciences*, 68(2):121101, 2025.

Huanyi Xie, Yubin Chen, Liangyu Wang, Lijie Hu, and Di Wang. Predicting LLM output length via entropy-guided representations. In *Proceedings of the 14th International Conference on Learning Representations (ICLR)*, pp. 67875–67905, 2026.

Dan Zhang, Sining Zhoubian, Ziniu Hu, Yisong Yue, Yuxiao Dong, and Jie Tang. ReST-MCTS*: LLM self-training via process reward guided tree search. In *Advances in Neural Information Processing Systems 37 (NeurIPS)*, pp. 64735–64772, 2024.

Zhen Zhang, Changyi Yang, Zijie Xia, Zhen Yang, Chengzhi Liu, Zhaotiao Weng, Yepeng Liu, Haobo Chen, Jin Pan, Chenyang Zhao, Yuheng Bu, Alkesh Patel, Zhe Gan, and Xin Eric Wang. Length value model: Scalable value pretraining for token-level length modeling. *ArXiv preprint arXiv:2604.27039*, 2026.

Haoyu Zheng, Yongqiang Zhang, Fangcheng Fu, Xiaokai Zhou, Hao Luo, Hongchao Zhu, Yuanyuan Zhu, Hao Wang, Xiao Yan, and Jiawei Jiang. Scheduling LLM inference with uncertainty-aware output length predictions. In *Proceedings of the 43rd International Conference on Machine Learning (ICML)*, pp. to appear, 2026.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Tianle Li, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zhuohan Li, Zi Lin, Eric Xing, Joseph E. Gonzalez, Ion Stoica, and Hao Zhang. LMSYS-Chat-1M: A large-scale real-world LLM conversation dataset. In *Proceedings of the 12th International Conference on Learning Representations (ICLR)*, pp. 22225–22257, 2024.

Zangwei Zheng, Xiaozhe Ren, Fuzhao Xue, Yang Luo, Xin Jiang, and Yang You. Response length perception and sequence scheduling: An LLM-empowered LLM inference pipeline. In *Advances in Neural Information Processing Systems 36 (NeurIPS)*, pp. 65517–65530, 2023.

Yinmin Zhong, Zili Zhang, Xiaoniu Song, Hanpeng Hu, Chao Jin, Bingyang Wu, Nuo Chen, Yukun Chen, Yu Zhou, Changyi Wan, Hongyu Zhou, Yimin Jiang, Yibo Zhu, and Daxin Jiang. StreamRL: Scalable, heterogeneous, and elastic RL for LLMs with disaggregated stream generation. *ArXiv preprint arXiv:2504.15930*, 2025.

Zhi-Hua Zhou. Learnability with time-sharing computational resource concerns. *National Science Review*, 11(10):nwae204, 2024.

Bowen Zuo and Yinglun Zhu. Strategic scaling of test-time compute: A bandit learning approach. In *Proceedings of the 14th International Conference on Learning Representations (ICLR)*, pp. 134105–134128, 2026.

A Related Work and Discussion

Length Prediction. Output length prediction has been studied for efficient batching and scheduling of LLM requests. Early works mainly predict the output length from the prompt. Zheng et al. (2023) proposed Perception in Advance (PiA), which elicits length estimates from the model itself. Jin et al. (2023) formulated length prediction as a classification problem in S^3, while Qiu et al. (2024) used a lightweight proxy predictor in SSJF. Cheng et al. (2024) proposed Magnus, which combines input length with semantic features for prediction. Instead of estimating lengths directly, Fu et al. (2024) learned the relative ordering of requests needed for scheduling. Subsequent works incorporate information observed during generation to update length predictions. Shahout et al. (2025) proposed TRAIL to predict probabilities over remaining-length intervals from model embeddings. Choi et al. (2025) developed ELIS, which encodes both the prompt and the observed response. Xie et al. (2026) proposed entropy-guided prompt representations in EGTP and further incorporated generated-token representations in its PLP extension. Length information in hidden states has also been studied through cross-layer graph forecasting (Piotrowski et al., 2025) and linear probing (Merzouk et al., 2026). More recently, Zhang et al. (2026) proposed LenVM to learn a transformed remaining-length return using dense trajectory supervision, and Wen et al. (2026) adapted speculative-decoding backbones for length forecasting. Related works have used reasoning-progress prediction to control reasoning length (Eisenstadt et al., 2025) and horizon-length prediction to improve code infilling (Ding et al., 2025b). Several works explicitly model length uncertainty. ProD (Wang et al., 2026) and TIE (Zheng et al., 2026) use repeated generations to learn prompt-conditioned length distributions. Budget Guidance (Li et al., 2026) models the remaining thinking length for budget control, while ZIP-RC (Manvi et al., 2026) predicts the joint distribution of final reward and remaining length. Our work studies how to acquire supervision for prefix-conditioned remaining-length distributions. Specifically, we sample multiple continuations from selected prefixes to refine their distributional targets, and use the states along these continuations to provide additional training supervision.

Sampling and Data Acquisition. Repeated sampling has been widely used to obtain information about alternative LLM outputs. Wang et al. (2023) proposed self-consistency, which aggregates multiple reasoning paths to improve answer selection. For supervision at intermediate states, Lightman et al. (2024) studied process supervision with step-level human feedback and active data selection. Wang et al. (2024) proposed Math-Shepherd to construct process labels from sampled continuations, while Zhang et al. (2024) developed ReST-MCTS* to collect training data through process-reward-guided tree search. Setlur et al. (2025) further evaluated a reasoning step by its effect on future success under a prover policy. These works use intermediate feedback or sampled continuations to supervise reasoning. In our setting, conditional continuations provide samples of the remaining length, from which we construct distributional targets for decoding states. The allocation of limited sampling computation has also received attention. Snell et al. (2025) studied compute-optimal test-time scaling by adapting computation to problem difficulty. Zuo & Zhu (2026) investigated bandit-based compute allocation, and Ding et al. (2025a) proposed BEST-Route to jointly select a model and a sample count. From a statistical perspective, adaptive stratified sampling allocates observations according to conditional variability (Carpentier & Munos, 2011; 2013). Nested conditional Monte Carlo studies the cost–variance trade-off of repeating simulations after shared prefixes (Dickmann & Schweizer, 2016), which is closely related to our use of multiple continuations from a decoding state. Our allocation accounts for both the variability of remaining length at a state and its influence on a shared predictor. We emphasize that branching is used only to acquire training supervision; at deployment, the predictor computes forecasts from observed states without sampling additional continuations.

B Proofs

Let $\mathcal{S}_0$ be the sigma-field generated by the ordinary data, shared history, and fixed planning quantities in Section 3.3. Let $\mathcal{S}$ additionally include the pilot outcomes and all randomness used to choose the production allocation. Conditional on $\mathcal{S}$, the counts are fixed and the direct production futures are independent within and across origins, with CDF $F_{\mathbf{v}}$ at origin $\mathbf{v}$. Throughout, $\mathbb{E}_{\mathrm{prod}}$ denotes expectation conditional on $\mathcal{S}$, and candidate sums range over $\mathcal{V}$. Write $L_{\mathbf{v}} = C - t_{\mathbf{v}}$. Remaining lengths belong to $\{1, \ldots, L_{\mathbf{v}}\}$, and CDF vectors are represented on $r = 0, \ldots, C-1$.

B.1 Conditional-Target Estimation

We first establish the error of the empirical conditional target and the unbiasedness of the pilot estimate.

Lemma 2 (Conditional-target error). *Let* $\varepsilon_{\mathbf{v}} = \bar{F}_{\mathbf{v}}^{\mathrm{prod}} - F_{\mathbf{v}}$. *Then*

$$\mathbb{E}_{\mathrm{prod}}\varepsilon_{\mathbf{v}} = 0, \qquad \mathbb{E}_{\mathrm{prod}}\|\varepsilon_{\mathbf{v}}\|_2^2 = \frac{A_{\mathbf{v}}}{m_{\mathbf{v}}}, \qquad A_{\mathbf{v}} = \sum_{r=0}^{C-1} F_{\mathbf{v}}(r)(1-F_{\mathbf{v}}(r)) \le \frac{L_{\mathbf{v}}-1}{4}. \tag{15}$$

Moreover, $A_{\mathbf{v}} \ge 0$ *and the estimate in Eq. (9) satisfies* $\mathbb{E}[\widehat{A}_{\mathbf{v}} \mid \mathcal{S}_0] = A_{\mathbf{v}}$.

Proof. For a fixed threshold r, the variables $\mathbb{1}_{\{R_{\mathbf{v},i}\le r\}}$ are conditionally independent Bernoulli variables with mean $F_{\mathbf{v}}(r)$. Hence,

$$\mathbb{E}_{\mathrm{prod}}\bar{F}_{\mathbf{v}}^{\mathrm{prod}}(r) = F_{\mathbf{v}}(r),$$
$$\mathbb{E}_{\mathrm{prod}}\varepsilon_{\mathbf{v}}(r)^2 = \frac{1}{m_{\mathbf{v}}^2}\sum_{i=1}^{m_{\mathbf{v}}}\mathbb{E}_{\mathrm{prod}}\left(\mathbb{1}_{\{R_{\mathbf{v},i}\le r\}} - F_{\mathbf{v}}(r)\right)^2 = \frac{F_{\mathbf{v}}(r)(1-F_{\mathbf{v}}(r))}{m_{\mathbf{v}}}.$$

Summing over r gives the squared-error identity in Eq. (15). Only thresholds $1,\dots,L_{\mathbf{v}}-1$ can contribute, and each term is at most $1/4$, proving the upper bound. This argument does not require independence between CDF coordinates.

For independent $R, R' \sim F_{\mathbf{v}}$, the integer-length identity

$$|R-R'| = \sum_{r=0}^{C-1}\left|\mathbb{1}_{\{R\le r\}} - \mathbb{1}_{\{R'\le r\}}\right|,$$

implies

$$\frac{1}{2}\mathbb{E}\,|R-R'| = \frac{1}{2}\sum_{r=0}^{C-1} 2F_{\mathbf{v}}(r)(1-F_{\mathbf{v}}(r)) = A_{\mathbf{v}}.$$

Applying this equality to the two pilot draws, which are independent with CDF $F_{\mathbf{v}}$ conditional on $\mathcal{S}_0$, yields $\mathbb{E}[\widehat{A}_{\mathbf{v}} \mid \mathcal{S}_0] = A_{\mathbf{v}}$. Thus we complete the proof. □

B.2 Proof of Theorem 1

Proof. Since $V \succeq \gamma I_d$, both ridge objectives have unique minimizers. Their normal equations give

$$\widehat{\Theta} = V^{-1}\sum_{\mathbf{v}\in\mathcal{V}} w_{\mathbf{v}}\mathbf{z}_{\mathbf{v}}(\bar{F}_{\mathbf{v}}^{\mathrm{prod}})^\top, \qquad \Theta^\circ = V^{-1}\sum_{\mathbf{v}\in\mathcal{V}} w_{\mathbf{v}}\mathbf{z}_{\mathbf{v}}F_{\mathbf{v}}^\top. \tag{16}$$

For an evaluation prefix $\mathbf{u}$, define $\alpha_{\mathbf{v}}(\mathbf{u}) = w_{\mathbf{v}}\mathbf{z}_{\mathbf{u}}^\top V^{-1}\mathbf{z}_{\mathbf{v}}$ and $\mathbf{e}_{\mathbf{u}} = (\Theta^\circ)^\top\mathbf{z}_{\mathbf{u}} - F_{\mathbf{u}}$. Subtracting the two solutions gives

$$\widehat{\Theta}^\top\mathbf{z}_{\mathbf{u}} - F_{\mathbf{u}} = \mathbf{e}_{\mathbf{u}} + \sum_{\mathbf{v}\in\mathcal{V}}\alpha_{\mathbf{v}}(\mathbf{u})\varepsilon_{\mathbf{v}}. \tag{17}$$

The coefficients and $\mathbf{e}_{\mathbf{u}}$ are fixed conditional on $\mathcal{S}$. Expanding the squared norm, we obtain

$$\mathbb{E}_{\mathrm{prod}}\left\|\widehat{\Theta}^\top\mathbf{z}_{\mathbf{u}} - F_{\mathbf{u}}\right\|_2^2 = \|\mathbf{e}_{\mathbf{u}}\|_2^2 + 2\sum_{\mathbf{v}\in\mathcal{V}}\alpha_{\mathbf{v}}(\mathbf{u})\left\langle\mathbf{e}_{\mathbf{u}}, \mathbb{E}_{\mathrm{prod}}\varepsilon_{\mathbf{v}}\right\rangle + \sum_{\mathbf{v},\mathbf{v}'\in\mathcal{V}}\alpha_{\mathbf{v}}(\mathbf{u})\alpha_{\mathbf{v}'}(\mathbf{u})\mathbb{E}_{\mathrm{prod}}\left\langle\varepsilon_{\mathbf{v}}, \varepsilon_{\mathbf{v}'}\right\rangle.$$

By Lemma 2, the linear term is zero. For $\mathbf{v} \ne \mathbf{v}'$, conditional independence gives $\mathbb{E}_{\mathrm{prod}}\left\langle\varepsilon_{\mathbf{v}}, \varepsilon_{\mathbf{v}'}\right\rangle = 0$. It follows that

$$\mathbb{E}_{\mathrm{prod}}\left\|\widehat{\Theta}^\top\mathbf{z}_{\mathbf{u}} - F_{\mathbf{u}}\right\|_2^2 = \|\mathbf{e}_{\mathbf{u}}\|_2^2 + \sum_{\mathbf{v}\in\mathcal{V}}\alpha_{\mathbf{v}}(\mathbf{u})^2\frac{A_{\mathbf{v}}}{m_{\mathbf{v}}}. \tag{18}$$

All expectations are finite because $V \succeq \gamma I_d$, the target vectors are bounded, and $\mathbb{E}_{\mathbf{u}\sim Q_{\mathrm{tr}}}\|\mathbf{z}_{\mathbf{u}}\|_2^2 < \infty$. Moreover, Eq. (8) gives

$$\mathbb{E}_{\mathbf{u}\sim Q_{\mathrm{tr}}}\alpha_{\mathbf{v}}(\mathbf{u})^2 = w_{\mathbf{v}}^2\mathbf{z}_{\mathbf{v}}^\top V^{-1}\mathbb{E}_{\mathbf{u}\sim Q_{\mathrm{tr}}}[\mathbf{z}_{\mathbf{u}}\mathbf{z}_{\mathbf{u}}^\top]V^{-1}\mathbf{z}_{\mathbf{v}} = w_{\mathbf{v}}^2\mathbf{z}_{\mathbf{v}}^\top V^{-1}\Sigma_{\mathrm{tr}}V^{-1}\mathbf{z}_{\mathbf{v}} = s_{\mathbf{v}}.$$

Taking expectation over $\mathbf{u}$ in Eq. (18) proves Eq. (13). The cancellation uses unbiased labels, not linear realizability or population optimality of Θ°. □

B.3 Proof of Lemma 1

Proof. Let $M = |\mathcal{V}|$, $\boldsymbol{m}^0 = (1)_{\mathbf{v}\in\mathcal{V}}$, and $K = \min\{B_{\text{prod}}, UM\} - M$. For each origin, define $d_{\mathbf{v},j} = \widehat{a}_{\mathbf{v}}/[j(j+1)]$, $j = 1, \ldots, U-1$. These marginal gains are nonnegative and nonincreasing in j. By telescoping,

$$\widehat{J}(\boldsymbol{m}^0) - \widehat{J}(\boldsymbol{m}) = \sum_{\mathbf{v}\in\mathcal{V}} \sum_{j=1}^{m_{\mathbf{v}}-1} \widehat{a}_{\mathbf{v}} \left(\frac{1}{j} - \frac{1}{j+1}\right) = \sum_{\mathbf{v}\in\mathcal{V}} \sum_{j=1}^{m_{\mathbf{v}}-1} d_{\mathbf{v},j}.$$

A feasible allocation selects an initial segment of each gain sequence, with at most K gains in total. At any step, the first unselected gain in a sequence is at least as large as every later gain in that sequence. Thus, selecting the largest feasible next gain selects a largest remaining gain globally while preserving the initial-segment constraint. The rule maximizes the sum of the selected gains and therefore minimizes $\widehat{J}$. Ties do not affect this argument, and stopping when all remaining gains are zero does not change the optimum. For $K = 0$, the initial allocation is the only feasible allocation.

To compare with $\boldsymbol{m}^\star$, write $\delta_{\mathbf{v}} = s_{\mathbf{v}}(A_{\mathbf{v}} - \widehat{A}_{\mathbf{v}})$. Optimality of $\widehat{\boldsymbol{m}}$ for $\widehat{J}$ gives

$$\begin{aligned} J(\widehat{\boldsymbol{m}}) - J(\boldsymbol{m}^\star) &= \widehat{J}(\widehat{\boldsymbol{m}}) - \widehat{J}(\boldsymbol{m}^\star) + \sum_{\mathbf{v}\in\mathcal{V}} \delta_{\mathbf{v}} \left(\frac{1}{\widehat{m}_{\mathbf{v}}} - \frac{1}{m^\star_{\mathbf{v}}}\right) \\ &\le \sum_{\mathbf{v}\in\mathcal{V}} s_{\mathbf{v}} \left|\widehat{A}_{\mathbf{v}} - A_{\mathbf{v}}\right| \left|\frac{1}{\widehat{m}_{\mathbf{v}}} - \frac{1}{m^\star_{\mathbf{v}}}\right| \\ &\le \sum_{\mathbf{v}\in\mathcal{V}} s_{\mathbf{v}} \left|\widehat{A}_{\mathbf{v}} - A_{\mathbf{v}}\right|, \end{aligned}$$

where the last inequality uses $\widehat{m}_{\mathbf{v}}, m^\star_{\mathbf{v}} \ge 1$. The lower bound follows from optimality of $\boldsymbol{m}^\star$. Thus we complete the proof. □

B.4 Relation to Training Supervision

Squared-CDF Error and CRPS. For a fixed CDF G and an independent $R \sim F_{\mathbf{u}}$, centering each indicator at its mean gives

$$\begin{aligned} \mathbb{E}_R \,\text{CRPS}(G, R) &= \sum_{r=0}^{C-1} \mathbb{E}_R\big(G(r) - \mathbb{1}_{\{R\le r\}}\big)^2 \\ &= \sum_{r=0}^{C-1} \big[(G(r) - F_{\mathbf{u}}(r))^2 + F_{\mathbf{u}}(r)(1 - F_{\mathbf{u}}(r))\big] \\ &= \|G - F_{\mathbf{u}}\|_2^2 + A_{\mathbf{u}}. \end{aligned} \tag{19}$$

Thus, squared-CDF error equals excess expected discrete CRPS. The same calculation under the empirical distribution gives

$$\frac{1}{n_{\mathbf{v}}} \sum_{R\in\mathcal{R}_{\mathbf{v}}} \text{CRPS}(G, R) = \big\|G - \bar{F}_{\mathbf{v}}\big\|_2^2 + \sum_{r=0}^{C-1} \bar{F}_{\mathbf{v}}(r)(1 - \bar{F}_{\mathbf{v}}(r)).$$

The second term does not depend on G, which establishes the empirical CDF-matching interpretation in Section 3.2. The algebra holds for any vector $G \in \mathbb{R}^C$, although the unconstrained ridge fit need not itself define a valid CDF.

Retained Observations. The production target $\bar{F}^{\text{prod}}_{\mathbf{v}}$ differs from the pooled target in Eq. (5). Let $n^{\text{old}}_{\mathbf{v}}$ be a fixed retained count and $F_{\mathbf{v},\text{old}}$ its empirical CDF. The pooled target is

$$T_{\mathbf{v}} = \frac{n^{\text{old}}_{\mathbf{v}} F_{\mathbf{v},\text{old}} + m_{\mathbf{v}} \bar{F}^{\text{prod}}_{\mathbf{v}}}{n^{\text{old}}_{\mathbf{v}} + m_{\mathbf{v}}}.$$

Conditional on the retained data, Lemma 2 yields

$$\mathbb{E}_{\text{prod}} T_{\mathbf{v}} - F_{\mathbf{v}} = \frac{n_{\mathbf{v}}^{\text{old}}}{n_{\mathbf{v}}^{\text{old}} + m_{\mathbf{v}}} (F_{\mathbf{v},\text{old}} - F_{\mathbf{v}}),$$
$$\mathbb{E}_{\text{prod}} \left\| T_{\mathbf{v}} - \mathbb{E}_{\text{prod}} T_{\mathbf{v}} \right\|_2^2 = \frac{m_{\mathbf{v}} A_{\mathbf{v}}}{(n_{\mathbf{v}}^{\text{old}} + m_{\mathbf{v}})^2}.$$

The first term is generally nonzero. Hence, substituting $n_{\mathbf{v}}^{\text{old}} + m_{\mathbf{v}}$ for $m_{\mathbf{v}}$ in Theorem 1 does not give the risk of pooled-label fitting.

Shared-Path Observations. For a fixed design, consider general targets $T_{\mathbf{v}}$ with finite conditional second moments. Define $\mu_{\mathbf{v}} = \mathbb{E}_{\text{prod}}(T_{\mathbf{v}} - F_{\mathbf{v}})$ and $\boldsymbol{\xi}_{\mathbf{v}} = T_{\mathbf{v}} - F_{\mathbf{v}} - \mu_{\mathbf{v}}$. Let $\widehat{\Theta}_T$ be the ridge fit to these targets with the same prefix weights. Using $\alpha_{\mathbf{v}}(\mathbf{u})$ and $\mathbf{e}_{\mathbf{u}}$ from Eq. (17), we have

$$\widehat{\Theta}_T^\top \mathbf{z}_{\mathbf{u}} - F_{\mathbf{u}} = \mathbf{e}_{\mathbf{u}} + \sum_{\mathbf{v} \in \mathcal{V}} \alpha_{\mathbf{v}}(\mathbf{u}) \mu_{\mathbf{v}} + \sum_{\mathbf{v} \in \mathcal{V}} \alpha_{\mathbf{v}}(\mathbf{u}) \boldsymbol{\xi}_{\mathbf{v}}.$$

The last sum has conditional mean zero. Expanding the squared norm and averaging therefore gives

$$\begin{aligned} \mathbb{E}_{\text{prod}} \mathcal{E}_{Q_{\text{tr}}}(\widehat{\Theta}_T) = \mathbb{E}_{\mathbf{u} \sim Q_{\text{tr}}} \left\| \mathbf{e}_{\mathbf{u}} + \sum_{\mathbf{v} \in \mathcal{V}} \alpha_{\mathbf{v}}(\mathbf{u}) \mu_{\mathbf{v}} \right\|_2^2 \\ + \sum_{\mathbf{v}, \mathbf{v}' \in \mathcal{V}} \mathbb{E}_{\mathbf{u} \sim Q_{\text{tr}}} [\alpha_{\mathbf{v}}(\mathbf{u}) \alpha_{\mathbf{v}'}(\mathbf{u})] \mathbb{E}_{\text{prod}} \left\langle \boldsymbol{\xi}_{\mathbf{v}}, \boldsymbol{\xi}_{\mathbf{v}'} \right\rangle. \end{aligned} \tag{20}$$

Labels reused along a shared path can have nonzero cross-origin terms. New path prefixes also make the feature design random; conditioning on their features does not in general restore independent, unbiased labels. The fixed-feature analysis therefore does not establish a gain from additional prefix coverage or from nonlinear refitting.

B.5 Information-Aware Planning Extension

The allocation in Section 3.3 keeps V fixed. We now consider a separate planning variant whose information matrix changes as continuations are assigned, following the experimental-design view of active regression and preference learning (Fontaine et al., 2021; Li et al., 2025). Its precision weights are used for planning and do not replace the prefix weights in the neural training loss.

For this extension only, assume $F_{\mathbf{u}} = \Theta_*^\top \mathbf{z}_{\mathbf{u}}$ at every candidate and Q_{tr}-almost every evaluation prefix, with $\|\Theta_*\|_{\text{F}} \le S$. Choose scales $\nu_{\mathbf{v}} > 0$ from the pilot, e.g., $\nu_{\mathbf{v}} = \max\{\widehat{A}_{\mathbf{v}}, \nu_{\min}\}$ for fixed $\nu_{\min} > 0$. Let $\widetilde{\Theta}_\nu(\boldsymbol{m})$ be the estimator in Eq. (12) with weights $m_{\mathbf{v}}/\nu_{\mathbf{v}}$ in place of $w_{\mathbf{v}}$. Define

$$H_\nu(\boldsymbol{m}) = \gamma I_d + \sum_{\mathbf{v} \in \mathcal{V}} \frac{m_{\mathbf{v}}}{\nu_{\mathbf{v}}} \mathbf{z}_{\mathbf{v}} \mathbf{z}_{\mathbf{v}}^\top, \qquad \Phi_\nu(\boldsymbol{m}) = \text{tr}\big(\Sigma_{\text{tr}} H_\nu(\boldsymbol{m})^{-1}\big). \tag{21}$$

Use the feasible set $\mathcal{M}_B$ from Eq. (10) and initialize $\boldsymbol{m}^0 = (1)_{\mathbf{v} \in \mathcal{V}}$. Writing $M = |\mathcal{V}|$, the planner makes $K = \min\{B_{\text{prod}}, UM\} - M$ virtual increments. At each increment, it chooses an origin with $m_{\mathbf{v}} < U$ maximizing

$$\Delta_\nu(\mathbf{v} \mid \boldsymbol{m}) = \frac{\mathbf{z}_{\mathbf{v}}^\top H_\nu(\boldsymbol{m})^{-1} \Sigma_{\text{tr}} H_\nu(\boldsymbol{m})^{-1} \mathbf{z}_{\mathbf{v}}}{\nu_{\mathbf{v}} + \mathbf{z}_{\mathbf{v}}^\top H_\nu(\boldsymbol{m})^{-1} \mathbf{z}_{\mathbf{v}}}, \tag{22}$$

and updates H_ν. Denote the resulting allocation by $\boldsymbol{m}^{\text{g}}$. All counts are committed before production; these matrix updates use no additional outcomes.

Theorem 2 (Information-aware continuation allocation)**.** *Under the shared linear model and conditional sampling assumptions, let*

$$\beta_\nu = \max\left\{ \gamma S^2, \max_{\mathbf{v} \in \mathcal{V}} \frac{A_{\mathbf{v}}}{\nu_{\mathbf{v}}} \right\}, \qquad \kappa = \left(1 + \max_{\mathbf{v} \in \mathcal{V}} \frac{\mathbf{z}_{\mathbf{v}}^\top H_\nu(\boldsymbol{m}^0)^{-1} \mathbf{z}_{\mathbf{v}}}{\nu_{\mathbf{v}}} \right)^{-1}.$$

Then $\kappa \ge 1/2$, and for every $\boldsymbol{m} \in \mathcal{M}_B$,

$$\mathbb{E}_{\text{prod}} \mathcal{E}_{Q_{\text{tr}}}(\widetilde{\Theta}_\nu(\boldsymbol{m})) \le \beta_\nu \Phi_\nu(\boldsymbol{m}). \tag{23}$$

For every feasible increment, $\Delta_\nu(\mathbf{v} \mid \boldsymbol{m}) = \Phi_\nu(\boldsymbol{m}) - \Phi_\nu(\boldsymbol{m} + \mathbf{e}_\mathbf{v})$, where $\mathbf{e}_\mathbf{v}$ is the coordinate vector for origin $\mathbf{v}$. Let $\boldsymbol{m}^\star_\nu$ minimize Φ_ν over $\mathcal{M}_B$. For $K \geq 1$,

$$\begin{aligned} \Phi_\nu(\boldsymbol{m}^{\mathrm{g}}) &\leq (1-\eta_K)\Phi_\nu(\boldsymbol{m}^0) + \eta_K \Phi_\nu(\boldsymbol{m}^\star_\nu), \\ \eta_K &= 1 - (1-\kappa/K)^K \geq 1 - e^{-\kappa}. \end{aligned} \tag{24}$$

For $K = 0$, $\boldsymbol{m}^{\mathrm{g}} = \boldsymbol{m}^0$ and we set $\eta_0 = 0$.

B.5.1 Proof of Theorem 2

The proof first bounds prediction error by Φ_ν, then derives its marginal decrease and a joint-gain inequality for greedy allocation. The latter follows the non-submodular analysis of Bian et al. (2017); we establish the required inequality directly.

Proof. Fix $\boldsymbol{m} \in \mathcal{M}_B$ and abbreviate $H = H_\nu(\boldsymbol{m})$. Define

$$D = \sum_{\mathbf{v}\in\mathcal{V}} \frac{m_\mathbf{v}}{\nu_\mathbf{v}} \mathbf{z}_\mathbf{v}\mathbf{z}_\mathbf{v}^\top = H - \gamma I_d, \qquad N = \sum_{\mathbf{v}\in\mathcal{V}} \frac{m_\mathbf{v} A_\mathbf{v}}{\nu_\mathbf{v}^2} \mathbf{z}_\mathbf{v}\mathbf{z}_\mathbf{v}^\top, \qquad \zeta_\nu = \max_{\mathbf{v}\in\mathcal{V}} \frac{A_\mathbf{v}}{\nu_\mathbf{v}}.$$

Under the shared linear model, the normal equations yield

$$\widetilde{\Theta}_\nu(\boldsymbol{m}) - \Theta_* = -\gamma H^{-1}\Theta_* + H^{-1} \sum_{\mathbf{v}\in\mathcal{V}} \frac{m_\mathbf{v}}{\nu_\mathbf{v}} \mathbf{z}_\mathbf{v} \varepsilon_\mathbf{v}^\top. \tag{25}$$

The second term has conditional mean zero. By independence across origins and Lemma 2,

$$\begin{aligned} \mathbb{E}_{\mathrm{prod}} \mathcal{E}_{Q_{\mathrm{tr}}}(\widetilde{\Theta}_\nu(\boldsymbol{m})) &= \gamma^2 \left\| \Sigma_{\mathrm{tr}}^{1/2} H^{-1} \Theta_* \right\|_{\mathrm{F}}^2 + \mathrm{tr}(\Sigma_{\mathrm{tr}} H^{-1} N H^{-1}) \\ &= \mathrm{tr}\big(\Sigma_{\mathrm{tr}} H^{-1} (\gamma^2 \Theta_* \Theta_*^\top + N) H^{-1}\big). \end{aligned} \tag{26}$$

Since $\Theta_*\Theta_*^\top \preceq S^2 I_d$ and $N \preceq \zeta_\nu D$, we have

$$\gamma^2 \Theta_*\Theta_*^\top + N \preceq \gamma^2 S^2 I_d + \zeta_\nu D \preceq \beta_\nu(\gamma I_d + D) = \beta_\nu H.$$

Substituting this inequality into Eq. (26) proves Eq. (23). The bias term is retained, and no independence between CDF coordinates is used.

For a feasible increment at $\mathbf{v}$, the Sherman–Morrison identity gives

$$(H + \mathbf{z}_\mathbf{v}\mathbf{z}_\mathbf{v}^\top/\nu_\mathbf{v})^{-1} = H^{-1} - \frac{H^{-1}\mathbf{z}_\mathbf{v}\mathbf{z}_\mathbf{v}^\top H^{-1}}{\nu_\mathbf{v} + \mathbf{z}_\mathbf{v}^\top H^{-1}\mathbf{z}_\mathbf{v}}. \tag{27}$$

Taking the trace against Σ_{tr} establishes $\Phi_\nu(\boldsymbol{m}) - \Phi_\nu(\boldsymbol{m} + \mathbf{e}_\mathbf{v}) = \Delta_\nu(\mathbf{v} \mid \boldsymbol{m}) \geq 0$. For an evaluation prefix $\mathbf{u}$, the same identity yields

$$\mathbf{z}_\mathbf{u}^\top H^{-1}\mathbf{z}_\mathbf{u} - \mathbf{z}_\mathbf{u}^\top (H + \mathbf{z}_\mathbf{v}\mathbf{z}_\mathbf{v}^\top/\nu_\mathbf{v})^{-1}\mathbf{z}_\mathbf{u} = \frac{(\mathbf{z}_\mathbf{u}^\top H^{-1}\mathbf{z}_\mathbf{v})^2}{\nu_\mathbf{v} + \mathbf{z}_\mathbf{v}^\top H^{-1}\mathbf{z}_\mathbf{v}}. \tag{28}$$

We next prove a joint-gain inequality. Let $H_0 = H_\nu(\boldsymbol{m}^0)$ and represent the allowed extra continuations by copies in $\mathcal{I} = \{(\mathbf{v}, j) : \mathbf{v} \in \mathcal{V},\ 1 \leq j \leq U-1\}$. For $i = (\mathbf{v}, j)$, set $\boldsymbol{b}_i = \mathbf{z}_\mathbf{v}/\sqrt{\nu_\mathbf{v}}$. For a subset $\mathcal{A} \subseteq \mathcal{I}$, define

$$H_\mathcal{A} = H_0 + \sum_{i\in\mathcal{A}} \boldsymbol{b}_i \boldsymbol{b}_i^\top, \qquad G(\mathcal{A}) = \mathrm{tr}(\Sigma_{\mathrm{tr}} H_0^{-1}) - \mathrm{tr}(\Sigma_{\mathrm{tr}} H_\mathcal{A}^{-1}).$$

A set of at most K copies represents a feasible count vector, and every feasible count vector has such a representation. Moreover, $G(\varnothing) = 0$ and G is nonnegative and nondecreasing.

For disjoint subsets $\mathcal{A}, \mathcal{J} \subseteq \mathcal{I}$, put $\boldsymbol{q}_i = H_\mathcal{A}^{-1/2}\boldsymbol{b}_i$, $P = H_\mathcal{A}^{-1/2}\Sigma_{\mathrm{tr}} H_\mathcal{A}^{-1/2}$, and $M_\mathcal{J} = \sum_{i\in\mathcal{J}} \boldsymbol{q}_i\boldsymbol{q}_i^\top$,

$$\begin{aligned} G(\mathcal{A}\cup\mathcal{J}) - G(\mathcal{A}) &= \mathrm{tr}\big(P[I_d - (I_d + M_\mathcal{J})^{-1}]\big) \\ &\leq \mathrm{tr}(PM_\mathcal{J}) = \sum_{i\in\mathcal{J}} \boldsymbol{q}_i^\top P \boldsymbol{q}_i, \end{aligned} \tag{29}$$

where the inequality follows from $0 \preceq I_d - (I_d + M_{\mathcal{J}})^{-1} \preceq M_{\mathcal{J}}$. For a singleton, Eq. (27) gives

$$G(\mathcal{A} \cup \{i\}) - G(\mathcal{A}) = \frac{\boldsymbol{q}_i^\top P \boldsymbol{q}_i}{1 + \|\boldsymbol{q}_i\|_2^2}.$$

Since $H_{\mathcal{A}} \succeq H_0$, $\|\boldsymbol{q}_i\|_2^2 \le \boldsymbol{b}_i^\top H_0^{-1} \boldsymbol{b}_i \le \kappa^{-1} - 1$. Together with Eq. (29), this implies

$$\sum_{i \in \mathcal{J}} \big(G(\mathcal{A} \cup \{i\}) - G(\mathcal{A})\big) \ge \kappa \big(G(\mathcal{A} \cup \mathcal{J}) - G(\mathcal{A})\big). \tag{30}$$

One continuation per origin is included in H_0, so $H_0 \succeq \mathbf{z}_{\mathbf{v}} \mathbf{z}_{\mathbf{v}}^\top / \nu_{\mathbf{v}}$ for every $\mathbf{v}$. Congruence by $H_0^{-1/2}$ yields $\mathbf{z}_{\mathbf{v}}^\top H_0^{-1} \mathbf{z}_{\mathbf{v}} / \nu_{\mathbf{v}} \le 1$, so $\kappa \ge 1/2$. This requires neither full-rank features nor submodularity.

Finally, suppose $K \ge 1$. Choose an optimal set $\mathcal{A}^\star$ of K copies, padding with zero-gain copies if necessary, and let $\mathcal{A}_k$ contain the first k greedy choices. By monotonicity and Eq. (30),

$$\begin{aligned} G(\mathcal{A}^\star) - G(\mathcal{A}_k) &\le G(\mathcal{A}_k \cup \mathcal{A}^\star) - G(\mathcal{A}_k) \\ &\le \kappa^{-1} \sum_{i \in \mathcal{A}^\star \setminus \mathcal{A}_k} \big(G(\mathcal{A}_k \cup \{i\}) - G(\mathcal{A}_k)\big) \\ &\le \frac{K}{\kappa} \big(G(\mathcal{A}_{k+1}) - G(\mathcal{A}_k)\big). \end{aligned}$$

The last inequality uses the largest feasible marginal selected by greedy. Rearranging and iterating for $k = 0, \dots, K-1$ gives

$$G(\mathcal{A}^\star) - G(\mathcal{A}_K) \le (1 - \kappa/K)^K G(\mathcal{A}^\star), \qquad G(\mathcal{A}_K) \ge \eta_K G(\mathcal{A}^\star).$$

Substituting the definition of G proves Eq. (24); the bound $\eta_K \ge 1 - e^{-\kappa}$ follows from $1 - x \le e^{-x}$. For $K = 0$, $\mathcal{M}_B$ contains only $\boldsymbol{m}^0$, proving the boundary case. Thus we complete the proof. □

Reference-Allocation Safeguard. Fix any reference allocation $\boldsymbol{m}^{\mathrm{ref}} \in \mathcal{M}_B$ before production. Between $\boldsymbol{m}^{\mathrm{g}}$ and $\boldsymbol{m}^{\mathrm{ref}}$, choose $\boldsymbol{m}^{\mathrm{act}}$ with smaller Φ_ν, using a fixed tie rule. By Theorem 2,

$$\begin{aligned} \mathbb{E}_{\mathrm{prod}} \mathcal{E}_{Q_{\mathrm{tr}}}(\widetilde{\Theta}_\nu(\boldsymbol{m}^{\mathrm{act}})) &\le \beta_\nu \Phi_\nu(\boldsymbol{m}^{\mathrm{act}}) \\ &= \beta_\nu \min\{\Phi_\nu(\boldsymbol{m}^{\mathrm{g}}), \Phi_\nu(\boldsymbol{m}^{\mathrm{ref}})\} \\ &\le \beta_\nu \min\big\{\Phi_\nu(\boldsymbol{m}^{\mathrm{ref}}), (1 - \eta_K)\Phi_\nu(\boldsymbol{m}^0) + \eta_K \Phi_\nu(\boldsymbol{m}_\nu^\star)\big\}. \end{aligned} \tag{31}$$

The comparison uses the same pilot, scales, regularizer, planning distribution, and production budget. It guarantees a bound no worse than the reference's bound, not an ordering of their actual risks.

Noise Scales and Scope. Lemma 2 implies

$$\beta_\nu \le \max\left\{\gamma S^2, \max_{\mathbf{v} \in \mathcal{V}} \frac{C - t_{\mathbf{v}} - 1}{4\nu_{\mathbf{v}}}\right\}.$$

A positive floor on $\nu_{\mathbf{v}}$ makes the criterion well defined, but does not ensure $\nu_{\mathbf{v}} \ge A_{\mathbf{v}}$. When this inequality does hold at every origin, the noise contribution ζ_ν is at most one. Regularization permits rank-deficient features; uncertainty in uncovered directions remains in the bias term of Eq. (26). Unlike Theorem 1, this extension assumes shared linear conditional CDFs and changes the planning estimator's weights. Neither result covers the full nonlinear objective or shared-path supervision. Both use unit continuation costs and production counts fixed before observing production outcomes.

C Experimental Details

This section gives a detailed description of the data construction, predictor implementations, continuation acquisition, and evaluation protocol used in the experiments.

C.1 Models and Datasets

Generation configuration. We use Qwen2.5-7B-Instruct and Llama-3-8B-Instruct for mathematical reasoning on GSM8K, code generation on MBPP, and conversation on first-turn LMSYS-Chat-1M requests. Table 3 summarizes the target configuration. Each source request is inserted into the corresponding chat template with a fixed system message, and the target model, tokenizer, template, and decoding policy are fixed within each setting. We generate one sequence per sample, without a minimum length, forced terminal token, extra stop string, or time-based stopping. An emitted EOS token contributes one token to the response length, and reaching the response cap C is treated as termination. A continuation from age t has remaining budget $C - t$, not C.

Table 3: Target-model and feature configurations. Hidden-state indices refer to the returned tuple, with index zero denoting the embedding output.

Configuration	Qwen	Llama
Temperature	0.8	0.8
Top-p / top-k	0.95 / disabled	0.95 / disabled
Repetition penalty	1.05	1.0
Response cap C	2048; Chat: 4096	2048
Target-state indices	$(2, 14, 26)$	$(2, 16, 30)$

Ordinary data and evaluation prefixes. Table 4 reports the ordinary collection D_0 and evaluation sizes. Evaluation prefixes are retained after 32, 64, 128, and 256 response tokens, provided that the original response remains active. We sample eight independent continuations from each retained prefix. Responses that terminate before age 32 remain in the root inventory but contribute no active-prefix score. Table 5 gives the resulting populations.

Table 4: Data sizes. Training samples belong to D_0 and exclude additional branches. Evaluation prompt counts include roots that terminate before the first evaluation age; each active prefix has eight evaluation futures.

		Training		Evaluation		
Model	Task	Prompts	Samples	Prompts	Prefixes	Futures
Qwen	Math	2751	6624	512	1861	14888
	Coding	374	1496	500	1909	15272
	Chat	1024	2304	512	1431	11448
Llama	Math	1024	2048	512	1189	9512
	Coding	374	1496	500	1408	11264
	Chat	1024	2048	512	1307	10456

Table 5: Active evaluation populations. Eligible prompts have at least one retained active prefix.

Setting	Eligible	Age 32	Age 64	Age 128	Age 256
Qwen Math	512	512	512	511	326
Qwen Coding	500	500	500	500	409
Qwen Chat	440	440	397	335	259
Llama Math	512	512	465	196	16
Llama Coding	489	489	432	379	108
Llama Chat	407	407	359	307	234

For Qwen Chat, responses that had already emitted EOS are retained, while previously capped responses are extended from their original token prefixes under the cap-4096 policy, preserving the ordinary sample identities. Capped outcomes are a small fraction of the collection and are retained in both training and evaluation.

C.2 Our Method

BranD. The causal auxiliary decoder receives the preceding target feature f_{i-1} and the current observed token embedding e_i. The residual history-attention module F uses multi-head attention with separate query and key/value layer normalizations and a zero-initialized output projection. The distribution head predicts an integer PMF on $1, \ldots, C - t$, and its median is the point readout. All features are restricted to the observed prefix; target states are obtained by prompt prefill followed by cached single-token processing.

Continuation acquisition. Branch continuations are drawn from candidate origins on the response prefix. Algorithm 1 summarizes the two-stage allocation: a small pilot per origin estimates dispersion and is used for planning only, after which production counts are committed and fresh fitting futures are collected. Allocation arms are compared under a matched total call budget rather than matched generated tokens or wall-clock time; the pilot cost is included in the budget. Table 6 reports the continuations collected in the transferred settings, including pilot costs.

Algorithm 1 BranD training with budgeted branching futures

Input: Ordinary data D_0, any fixed shared history, target policy, response cap C, call budget B, per-origin cap U

Fit the initial predictor ψ_0 on the base collection.
Fix candidate origins $\mathcal{V}$, prefix weights, and Q_{tr}; set $M = |\mathcal{V}|$ and $B_{\mathrm{prod}} = B - 2M$.
if $M = 0$ or $B_{\mathrm{prod}} < M$ **then**
    return ψ_0
end if
Freeze the planning representation and compute sensitivities s_v.
Draw two independent pilot futures per origin and set $\widehat{A}_v = |R^{\mathrm{pil}}_{v,1} - R^{\mathrm{pil}}_{v,2}|/2$.
Set $\widehat{a}_v = s_v \widehat{A}_v$ and $m_v = 1$ for all v.
while $\sum_v m_v < B_{\mathrm{prod}}$ and a positive feasible gain remains **do**
    Select v with $m_v < U$ maximizing $\widehat{a}_v/[m_v(m_v + 1)]$, using a fixed tie rule.
    Set $m_v \leftarrow m_v + 1$.
end while
Commit all counts and draw m_v independent production futures from each origin, with remaining cap $C - t_v$.
Add origin and retained descendant observations, pooling exact prefixes while preserving independent sample identities.
Refit from ψ_0 on the augmented collection, excluding pilot labels, selecting the checkpoint on held-out development data.
return The refit predictor and realized acquisition counts.

Prefix identity and path supervision. A prefix is identified by the target policy, prompt tokens, and complete observed response tokens. Independent samples remain distinct outcomes even when their token sequences coincide. A continuation from age t_v with remaining length R contributes label R at its origin and $R - (s - t_v) > 0$ at retained active descendants of age s. Descendants are retained on the 32-token grid strictly after the origin. Revisiting an existing prefix refines its target without duplicating its node weight.

Table 6: Additional continuation acquisition. Tokens count newly generated continuation tokens.

	Origins	Pilot		Production	
Setting		Samples	Tokens	Samples	Tokens
Qwen Coding	78	156	46642	468	126134
Qwen Chat	73	146	85361	292	143432
Llama Math	54	108	10631	324	34180
Llama Coding	59	118	16090	354	55440
Llama Chat	51	102	34130	52	13854

C.3 Baseline Implementations

Empirical. This featureless baseline estimates the empirical distribution of total response lengths in D_0. At age t, it conditions on $L > t$ and shifts by t to obtain a remaining-length distribution. It uses survival and elapsed length, but not content. Its readout is the conditional median.

OUTLETS. We use the LP-only implementation (Wen et al., 2026), retaining its scalar head, length objective, and native point readout. BranD's residual module, PMF head, distributional loss, and branch labels are not added to this external baseline. Each external baseline is trained on the complete ordinary trajectory collection D_0 of its setting.

ProD with survival conditioning. We use the distribution-estimation variant of ProD (Wang et al., 2026), which combines a histogram predictor, tail-CDF predictors, and fusion, and conditions the predicted total-length distribution on survival.

Table 7: Qwen Math prediction scores. Native MAE uses the declared point readout; median-MAE uses the PMF median. Baselines use D_0 only, while BranD also uses branch supervision. NLL is in nats; other scores are in tokens. A dash denotes an inapplicable integer-PMF score.

Method	CRPS	Native MAE	Median-MAE	NLL
Empirical	44.402	62.654	62.654	∞
ProD	33.763	46.429	46.430	5.598
TRAIL	40.738	52.923	55.873	5.894
PLP	52.738	43.695	60.836	∞
OUTLETS	35.371	35.371	–	–
BranD	24.990	34.137	34.137	5.155

TRAIL. The predictor uses target hidden states with a small ReLU classifier over a fixed set of length bins, trained with hard-label cross-entropy, and its point readout is the posterior expectation of bin centers. Our implemented recurrence follows the paper's printed equations,

$$a_0 = p_0, \qquad a_t = T a_{t-1}, \qquad q_t \propto a_t \odot p_t,$$

with a length-shifting transition matrix T. The coarse bins should also be considered when interpreting this baseline.

PLP. We build on the public EGTP release (Xie et al., 2026), retaining its head and entropy-weighted pooling over prompt states with the frozen target LM head. To extend it to the variable-length setting, we combine prompt and response features through a masked mean,

$$z_t = \frac{h_{\text{prompt}} + \sum_{j=1}^{t} h'_j}{t+1}.$$

The objective combines a soft-target KL term with a raw-token squared-error term.

Probability reconstruction. For binned predictors, native probability mass is distributed uniformly over the integers in each complete bin, restricted to $1 \le r \le C - t$, and renormalized. No additional calibration or probability floor is fitted. Native point outputs remain unchanged; they are not replaced by medians of reconstructed PMFs. A scalar output is evaluated as an exact, potentially non-integer point mass for CRPS.

C.4 Evaluation and Statistical Procedures

Prediction scores. Let $\mathcal{X}^{+}_{\text{eval}}$ contain prompts with at least one active evaluation prefix, and let $\mathcal{T}_X$ be the retained ages for prompt X. For a point output a_v, predicted CDF $\widehat{F}_\psi(\cdot \mid v)$, and $K_v = 8$

independent evaluation futures, define

$$e_v^{\mathrm{MAE}} = \frac{1}{K_v}\sum_{j=1}^{K_v} |a_v - R_{v,j}|,$$
$$e_v^{\mathrm{CRPS}} = \frac{1}{K_v}\sum_{j=1}^{K_v}\sum_{r=0}^{C-1}\left(\widehat{F}_\psi(r \mid v) - \mathbf{1}\{R_{v,j} \le r\}\right)^2. \tag{32}$$

For either score, we average first within prefixes, then within prompts:

$$\mathrm{Score} = \frac{1}{|\mathcal{X}^+_{\mathrm{eval}}|}\sum_{X\in\mathcal{X}^+_{\mathrm{eval}}}\frac{1}{|\mathcal{T}_X|}\sum_{t\in\mathcal{T}_X} e_{v_{X,t}}. \tag{33}$$

Reported CRPS is in tokens and is not divided by C; the scaling used in training does not rescale evaluation. At a fixed age, all methods use the same surviving prompts, and terminated responses are not padded with zero-error observations. Root predictions are evaluated separately, including prompts without later active prefixes. For a scalar output, continuous Dirac CRPS equals absolute error without rounding onto the integer grid. For integer PMFs, negative log-likelihood (NLL) uses the same prompt weighting; genuine zero probabilities yield infinite NLL and are not floored.

D Additional Experimental Results

We report distribution diagnostics, component comparisons, conditional-target estimation, and the exploratory serving study. Unless stated otherwise, prediction scores follow Appendix C.4.

D.1 Prediction Dynamics and Distribution Quality

Point readouts and probability support. Table 7 compares native point outputs, PMF medians, and probability scores on the 512-prompt Qwen Math cohort. Native and reconstructed-median readouts need not agree for binned baselines; in particular, PLP's native point error is lower than the error of its reconstructed median, which is why the two readouts are reported separately.

Empirical and the finite-interval PLP reconstruction assign zero probability to some observed future events, which explains their infinite NLL.

D.2 Conditional-Target Estimation

Configuration. This experiment studies empirical conditional targets, rather than the test error of a trained neural predictor. The Qwen observation collection contains 18 fixed conditions from four prompts. Each condition has 32 estimation futures and an independent 128-future reference. We form nested subsets of sizes $n \in \{1, 2, 4, 8, 16, 32\}$ by resampling each estimation pool, then average conditions within prompts and prompts equally.

Table 8: Squared-CDF distance to independent finite reference distributions on the four-prompt observation collection. Resampled subsets reuse the same estimation pools and are not independent acquisitions.

Estimation futures n	1	2	4	8	16	32
Squared-CDF distance	8.630	4.371	2.285	1.264	0.739	0.475

Distance and results. For length samples X and Y, the unbinned distance is

$$\|F_X - F_Y\|_2^2 = \frac{1}{|X||Y|}\sum_{x\in X}\sum_{y\in Y}|x-y| - \frac{1}{2|X|^2}\sum_{x,x'\in X}|x-x'| - \frac{1}{2|Y|^2}\sum_{y,y'\in Y}|y-y'|. \tag{34}$$

Table 8 shows that more futures improve target estimation on these conditions. The observed decrease reflects estimation of the empirical conditional target and does not imply the same rate for a fitted predictor's test CRPS.