ACTIVE AGENTIC LEARNING

Anonymous authors
Paper under double-blind review

ABSTRACT

Training capable large language model agents requires costly trajectory annotations that contain multi-step reasoning, tool calls, and environment interactions. Active learning can reduce this burden by prioritizing informative trajectories, but conventional uncertainty scores are poorly matched to agentic data: consequential decisions are diluted by predictable transcript tokens, while environment observations can vary independently of the model's pre-call confidence. We formulate **Active Agentic Learning** and propose **Pseudo-Influence**, a trajectory-valuation criterion that directly estimates training impact through pseudo-label perturbation rather than predictive uncertainty. Our efficient estimator provides theoretical guarantees without requiring a convex objective or inverse-Hessian computation. We evaluate Qwen3-8B and Qwen3-32B over five acquisition rounds in two complementary settings: command-line interaction on TERMINAL-BENCH 2.0 and tool use on ACEBENCH-AGENT. Across all 16 evaluation points before full-pool training, the best Pseudo-Influence variant ranks first and exceeds the strongest non-Pseudo-Influence baseline by 1.4–5.3 pass@1 points, with a mean gain of 3.1 points. In the first acquisition round, combining Pseudo-Influence with diversity improves over random selection by 4.8 and 5.5 points on TERMINAL-BENCH 2.0, and by 6.7 points for both model scales on ACEBENCH-AGENT. These results indicate that training-impact valuation and diversity are complementary signals for annotation-efficient agent learning.

1 INTRODUCTION

The rapid advancement of large language models (LLMs) has catalyzed a paradigm shift from passive question-answering systems to *agentic AI* autonomous agents capable of multi-step reasoning, tool invocation, and environment interaction to accomplish complex tasks (Xi et al., 2023; Wang et al., 2023). This transition has unlocked unprecedented capabilities in domains ranging from software development (Jimenez et al., 2024) to scientific discovery (Boiko et al., 2023). However, the training of such agentic systems comes at a staggering cost. Unlike conventional supervised fine-tuning on static QA pairs, agentic training requires large-scale *trajectory annotations*, where each sample consists of a multi-step sequence of observations, reasoning steps, and tool calls. The human effort and computational resources needed to annotate these trajectories are orders of magnitude higher than those for standard LLM post-training, creating a critical bottleneck for scaling agentic AI.

Active learning (AL) (Lewis & Gale, 1994; Cohn et al., 1994; Atlas et al., 1989; Angluin, 1988) offers a principled solution to this challenge by iteratively selecting the most informative samples from a large unlabeled pool for annotation, thereby maximizing model improvement per annotation dollar. The core of any AL method lies in its *sample valuation* criterion: how to quantify the potential contribution of an unlabeled sample to model training. In traditional settings such as image classification and LLM QA post-training, this problem has been extensively studied through two dominant paradigms: *uncertainty-based* methods, which prioritize samples where the model is least confident (Cody Coleman et al., 2019; Roth & Small, 2006; Gal et al., 2017), and *diversity-based* methods, which seek to cover the data distribution through clustering or core-set selection (Bodó et al., 2011; Ash et al., 2020; Ozan Sener & Silvio Savarese, 2018). Diversity-based is an *orthogonal dimension* to sample valuation; it governs *which* samples to cover, while valuation determines *how informative* each sample is. These two dimensions are not mutually exclusive and can be combined naturally, as we show in our experiments. **However, uncertainty-based valuation faces fundamental obstacles for agentic scenarios.**

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.

Metrics such as entropy, perplexity (PPL), and their variants fail to capture the true informativeness of agentic trajectories because of two intrinsic mismatches: (1) *decision-token dilution*, where a few consequential actions are averaged with many predictable characters; and (2) *exogenous environment variation*, where tool observations vary independently of the model's confidence. We provide a detailed empirical analysis and concrete illustrations of these limitations in Sec. 2.

To address the valuation challenge in agentic active learning, we propose **Pseudo-Influence**, a novel sample valuation framework that directly measures the *training impact* of an unlabeled trajectory by estimating how its pseudo-labeled version would influence the model's learning dynamics. Computing the exact influence is intractable due to exponential complexity in the sample space (Hammoudeh & Lowd, 2022). We therefore devise an efficient estimator with theoretical guarantees that overcomes two key limitations of classical influence functions (Koh & Liang, 2017): (1) it does not require the objective function to be convex with respect to network parameters, and (2) it avoids the computationally prohibitive inverse Hessian computation, enabling scalable application to large-scale agentic datasets. Empirically, across two agentic benchmarks, two model scales, and all 16 acquisition points before full-pool training, the best Pseudo-Influence variant ranks first and improves over the strongest non-Pseudo-Influence baseline by 1.4–5.3 pass@1 points, with an average gain of 3.1 points. In summary, our contributions are threefold:

- We systematically identify and empirically analyze the failure modes of traditional uncertainty-based metrics in agentic scenarios, revealing two intrinsic mismatches between scalar confidence estimation and interactive trajectory valuation.
- We propose Pseudo-Influence, the first sample valuation criterion *specifically designed for agentic scenarios*. Instead of relying on uncertainty estimation, it directly quantifies the *training impact* of each unlabeled trajectory via pseudo-label perturbation, with an efficient estimator that bypasses the convexity assumption and inverse Hessian computation of classical influence functions.
- We evaluate Qwen3-8B and Qwen3-32B on command-line interaction and tool use. Pseudo-Influence leads every pre-full-data evaluation point, while its combination with diversity improves over random selection by 4.8–6.7 pass@1 points in the first acquisition round, indicating strong annotation efficiency across tasks and model scales.

The detailed literature review is placed in the appendix. Please refer to the Appendix. A.

2 Why Uncertainty Fails for Agentic Trajectories

Uncertainty-based active learning assumes that low model confidence identifies samples that are especially informative for training. We first test this assumption with Bits Per Character (BPC), a character-normalized likelihood measure that reduces sensitivity to tokenizer granularity. For an answer x represented by model tokens $t_1, \ldots, t_{N(x)}$, we define

$$\mathrm{BPC}_\theta(x) = -\frac{1}{C(x)} \sum_{i=1}^{N(x)} \log_2 p_\theta(t_i \mid t_{<i}), \tag{1}$$

where $N(x)$ is the number of tokens, $C(x)$ is the number of characters in the detokenized answer, and p_θ is the next-token distribution of model θ. A lower BPC indicates higher confidence per character.

We evaluate nine language models on the official Terminal-Bench original tasks[1] (Merrill et al., 2026). For each model, we compute BPC on the official reference solutions and average it across tasks. We independently execute the model on the same tasks and measure success with the official task verifiers. Thus, the reported scores are our measurements rather than values copied from the public leaderboard.

Figure 1 reveals a coarse negative trend, but not a sufficiently precise relation for fine-grained valuation. A linear fit explains only 31.9% of the score variance. More importantly, the model with the lowest BPC, Kimi K2.6, ranks fifth by task score, while DeepSeek-Flash ranks first by task score

[1] https://github.com/harbor-framework/terminal-bench-1/tree/main/original-tasks

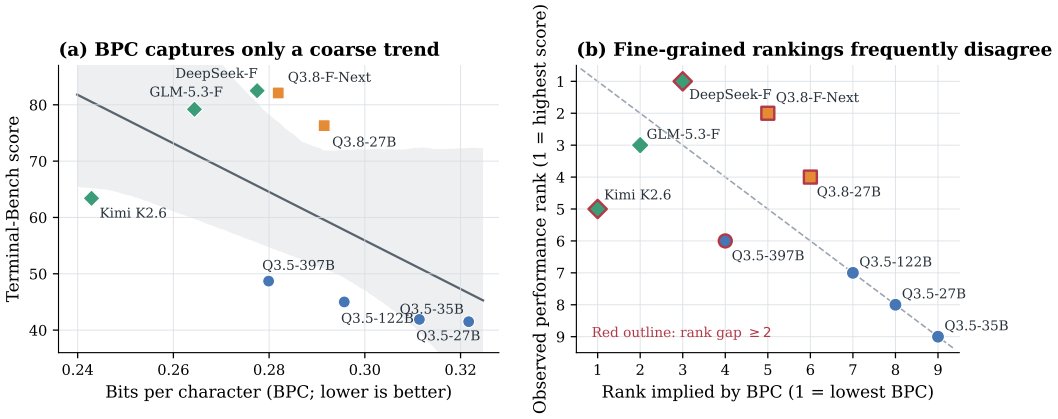


Figure 1: **BPC provides only a coarse proxy for agentic capability.** (a) Lower BPC is directionally associated with a higher TERMINAL-BENCH score, but the relationship is imprecise across nine models: Pearson $r = -0.564$ ($p = 0.113$), Spearman $\rho = -0.683$ ($p = 0.042$), and linear $R^2 = 0.319$. The shaded region is a 95% bootstrap confidence interval for the linear fit. (b) Ranking models by BPC produces substantial disagreements with their observed performance ranking. A red outline marks a rank gap of at least two positions. Lower BPC and higher task score are better.

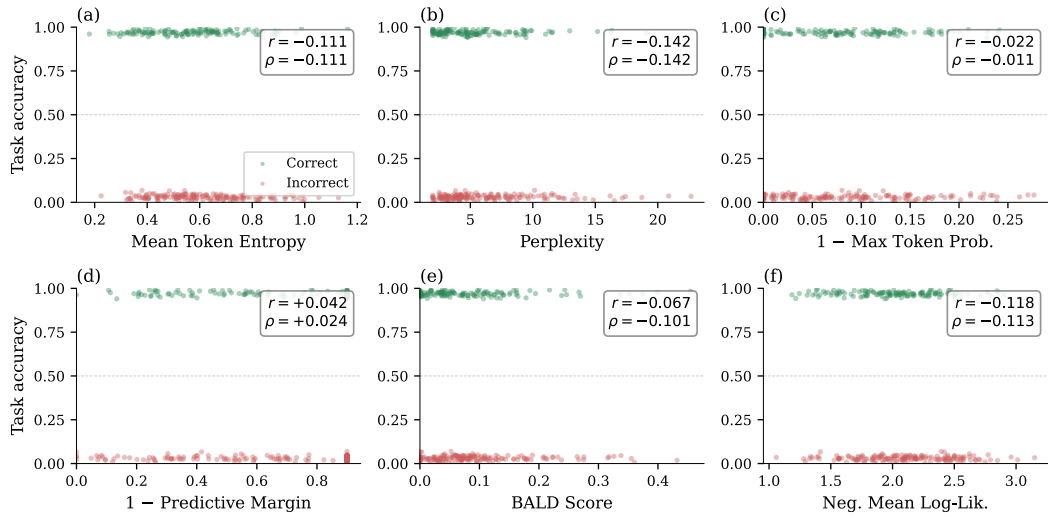


Figure 2: **Illustrative per-task view of weak uncertainty separation.** We collect 300 binary task outcomes and six sequence-level uncertainty metrics to visualize how correct and incorrect tasks can occupy nearly the same uncertainty range. The task-accuracy axis is fixed to $[0, 1]$, with small within-class jitter added only to reveal point density, while each uncertainty axis spans its range with a 5% margin. Pearson r and Spearman ρ are displayed in each panel. These points are a controlled illustration, not additional measured benchmark results. The uncertainties in (c) and (d) are re-scaled for better visualization.

but only third by BPC. The result does not imply that likelihood is entirely uninformative. Instead, it shows that a scalar likelihood average omits factors that determine whether an agent completes an interactive task.

The visualization in Figure 2 makes the observed failure mode visually concrete: when sequence-level scores are dominated by predictable transcript content and outcomes also depend on external observations, successful and failed tasks need not separate along a scalar uncertainty axis. We next isolate these two mechanisms with worked examples.

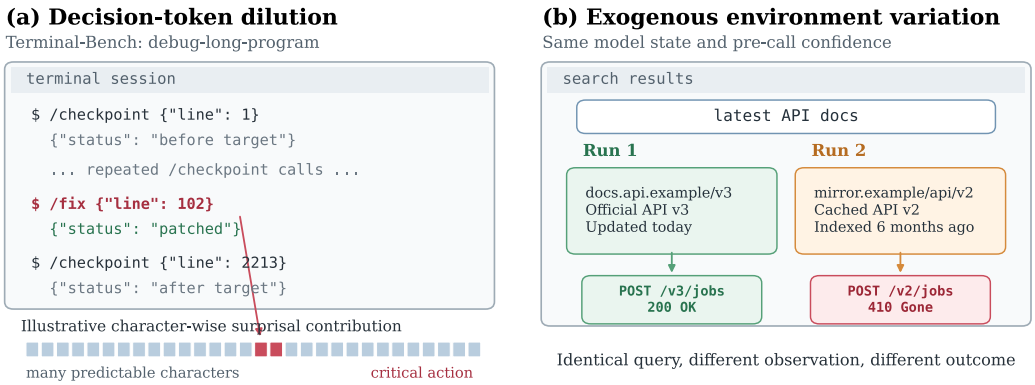


Figure 3: **Concrete illustrations of two mismatches between scalar uncertainty and agentic outcomes.** Both panels are illustrative renderings rather than logged model runs. (a) A TERMINAL-BENCH debugging trajectory contains many predictable command characters but only a few decision-critical repair actions. The strip visualizes how a local high-surprisal decision can be diluted by the full transcript average. (b) The same search query can expose a current API document or a stale cached page, leading to different actions despite the same model state and pre-call confidence.

(1) Decision-token dilution. An agentic trajectory mixes a small number of consequential decisions with large amounts of predictable syntax, tool output, and formatting. Averaging likelihood over the full sequence therefore gives most weight to characters that are easy to predict but weakly related to task success. In the `debug-long-program` example shown in Figure 3(a), the reference trajectory contains many repetitive `/checkpoint` calls, while successful debugging hinges on two `/fix` actions at lines 102 and 2345. The terminal view shows one such repair at line 102, and the strip below illustrates its localized contribution to sequence-level surprisal. A model can assign high likelihood to the repeated command template while choosing the wrong repair location, leaving aggregate BPC low even when the decisive action fails.

(2) Exogenous environment variation. Trajectory outcomes also depend on observations that are not generated by the language model. Search indices change, APIs return time-dependent content, and tools may expose different environment states. Figure 3(b) illustrates two calls to the same query from the same model state. One result exposes current API documentation and supports a valid endpoint, while the other exposes a stale mirror and suggests an obsolete endpoint. The model can have identical confidence before both calls but receive observations that lead to different outcomes. Because BPC only measures likelihood under the model, it cannot represent this external source of outcome variation.

3 METHOD

In the following, we will first present our pseudo-influence metric in Sec. 3.1. Then, we will show the proposed efficient estimator for the metric along with the complexity analysis and error guarantees in Sec. 3.2. Furthermore, we will introduce the active learning pipeline with the proposed metric in 3.3.

3.1 DEFINITION OF PSEUDO-INFLUENCE

Let $\mathcal{D} = \{z_i\}_{i=1}^N$ be the current labeled trajectory set and let $z = (x, \tilde{y})$ be a candidate trajectory with a pseudo-label $\tilde{y}$. We define $\ell(z_i; \theta)$ as the mean loss over the valid tokens of trajectory z_i and assign equal weight to trajectories. The corresponding empirical objectives are

$$F_{\mathcal{D}}(\theta) = \frac{1}{N} \sum_{i=1}^{N} \ell(z_i; \theta), \qquad F_{\mathcal{D}+z}(\theta) = \frac{N F_{\mathcal{D}}(\theta) + \ell(z; \theta)}{N+1}. \tag{2}$$

For the theoretical analysis, we consider coupled full-batch gradient-descent trajectories. Starting from the same initialization θ^0, they follow

$$\theta_{\mathcal{D}}^{t+1} = \theta_{\mathcal{D}}^t - \eta_t \nabla F_{\mathcal{D}}(\theta_{\mathcal{D}}^t), \quad \theta_{\mathcal{D}+z}^{t+1} = \theta_{\mathcal{D}+z}^t - \eta_t \nabla F_{\mathcal{D}+z}(\theta_{\mathcal{D}+z}^t), \tag{3}$$

for $t = 0, \ldots, T-1$. Let Π be a fixed linear projection with $\|\Pi\|_{\mathrm{op}} \leq 1$ onto the parameter block used for scoring. Taking Π as the identity gives full-parameter influence, while our implementation uses the LM-head block for efficiency.

Definition 3.1 (Finite-horizon pseudo-influence)**.** The exact pseudo-influence of z under a training budget of T steps is

$$\mathcal{I}_{p,T}^{\Pi}(z) = \left\| \Pi \left(\theta_{\mathcal{D}+z}^T - \theta_{\mathcal{D}}^T \right) \right\|_2 . \tag{4}$$

It is the parameter displacement caused by inserting the pseudo-labeled trajectory under the same initialization and the gradient-descent schedule in Eq. (3). This finite-horizon target matches brute-force retraining under that update rule and approaches the corresponding converged score when both trajectories converge.

3.2 Efficient Approximation

Computing Eq. (4) separately for every candidate is prohibitively expensive. At a saved base checkpoint $\theta_{\mathcal{D}}^t$, define

$$G_{\mathcal{D}}^t = \nabla F_{\mathcal{D}}(\theta_{\mathcal{D}}^t), \quad G_z^t = \nabla \ell(z; \theta_{\mathcal{D}}^t), \quad h_z^t = \frac{G_z^t - G_{\mathcal{D}}^t}{N+1}. \tag{5}$$

The last equality is not an approximation: by Eq. (2), $h_z^t = \nabla F_{\mathcal{D}+z}(\theta_{\mathcal{D}}^t) - \nabla F_{\mathcal{D}}(\theta_{\mathcal{D}}^t)$. Let $\eta_s = \sum_{t=0}^{T-1} \eta_t$ and $P(t) = \eta_t / \eta_s$. Replacing gradients on the unavailable perturbed trajectory by gradients at the saved base checkpoints gives

$$\bar{\Delta}_T(z) = -\sum_{t=0}^{T-1} \eta_t \Pi h_z^t = -\eta_s \, \mathbb{E}_{t \sim P} \left[\Pi h_z^t \right]. \tag{6}$$

In practice, we sample m checkpoints $t_1, \ldots, t_m \overset{\text{i.i.d.}}{\sim} P$ and use mini-batch gradient estimates $\widetilde{h}_z^{t_j}$:

$$\widehat{\mathcal{I}}_{p,m}^{\Pi}(z) = \left\| -\frac{\eta_s}{m} \sum_{j=1}^{m} \Pi \widetilde{h}_z^{t_j} \right\|_2 . \tag{7}$$

Proposition 1 (Exact trajectory decomposition). Let $\Delta_z^t = \theta_{\mathcal{D}+z}^t - \theta_{\mathcal{D}}^t$ and $a_z^t = \nabla F_{\mathcal{D}+z}(\theta_{\mathcal{D}+z}^t) - \nabla F_{\mathcal{D}+z}(\theta_{\mathcal{D}}^t)$. Then

$$\Pi \Delta_z^T = \bar{\Delta}_T(z) - \sum_{t=0}^{T-1} \eta_t \Pi a_z^t. \tag{8}$$

Consequently, Eq. (6) is obtained by freezing the candidate-augmented gradient to the saved base trajectory, and Eq. (7) is its Monte Carlo implementation.

Appendix B derives Proposition 1 and the approximation bound.

The perturbation h_z^t contrasts the candidate gradient with the mean gradient already induced by $\mathcal{D}$. If these gradients are similar throughout training, inserting z changes the update trajectory little. A large accumulated difference instead indicates that the candidate contributes a direction not already represented by the labeled set. Taking the norm after accumulating signed perturbations is important because it preserves cancellation across checkpoints before assigning a scalar score.

3.2.1 Approximation error guarantee

The approximation has three distinct error sources: evaluating the candidate perturbation on the base rather than the perturbed training path, sampling only m checkpoints, and estimating full-data gradients with mini-batches.

Algorithm 1 Active learning with pseudo-influence.

Input: Labeled trajectories $\mathcal{D}_l$, unlabeled tasks $\mathcal{D}_u$, initial parameters θ^0, annotation budget B, acquisition size δ, and checkpoint count m.
while $|\mathcal{D}_l| < B$ **do**
  // Train and cache the base trajectory.
  Train the surrogate model on $\mathcal{D}_l$ for T steps, saving $\theta^t_{\mathcal{D}_l}$, η_t, and mini-batch estimates of $G^t_{\mathcal{D}_l}$.
  Sample $t_1, \dots, t_m \overset{\text{i.i.d.}}{\sim} P(t) = \eta_t/\eta_s$.
  // Construct and score pseudo-labeled trajectories.
  for $x \in \mathcal{D}_u$ **do**
    Roll out the surrogate model to obtain $z = (x, \tilde{y})$.
    Compute $\widehat{\mathcal{I}}^{\Pi}_{p,m}(z)$ using Eq. (7).
  end for
  // Select, annotate, and update.
  Set $q = \min\{\delta, B - |\mathcal{D}_l|\}$ and select the task set $\mathcal{X}_s = \arg\max_{\mathcal{X} \subseteq \mathcal{D}_u, |\mathcal{X}|=q} \sum_{x \in \mathcal{X}} \widehat{\mathcal{I}}^{\Pi}_{p,m}(z_x)$.
  Obtain oracle trajectories $\mathcal{Z}_s$ for $\mathcal{X}_s$, update $\mathcal{D}_l \leftarrow \mathcal{D}_l \cup \mathcal{Z}_s$, and set $\mathcal{D}_u \leftarrow \mathcal{D}_u \setminus \mathcal{X}_s$.
end while
Output: The labeled trajectory set $\mathcal{D}_l$.

Theorem 3.2 (Finite-horizon approximation bound)**.** *Assume that* $\|\nabla F_{\mathcal{D}+z}(\theta^t_{\mathcal{D}+z}) - \nabla F_{\mathcal{D}+z}(\theta^t_{\mathcal{D}})\|_2 \le \beta\|\theta^t_{\mathcal{D}+z} - \theta^t_{\mathcal{D}}\|_2$, *that* $\|\nabla F_{\mathcal{D}}(\theta^t_{\mathcal{D}})\|_2 \le G$ *and* $\|\nabla \ell(z; \theta^t_{\mathcal{D}})\|_2 \le G$, *and that* $\eta_t \ge 0$ *with* $\eta_s > 0$. *Let* $\mu_z = \mathbb{E}_{t\sim P}[\Pi h^t_z]$ *and* $\sigma^2_z = \mathbb{E}_{t\sim P}\|\Pi h^t_z - \mu_z\|^2_2$. *Suppose the mini-batch estimate obeys* $\|\Pi(\widetilde{h}^t_z - h^t_z)\|_2 \le \varepsilon_B$ *almost surely for every sampled* t. *Then, for any* $0 < \rho < 1$, *with probability at least* $1 - \rho$ *over* m *independently sampled checkpoints,*

$$\left|\mathcal{I}^{\Pi}_{p,T}(z) - \widehat{\mathcal{I}}^{\Pi}_{p,m}(z)\right| \le \underbrace{\frac{2G}{\beta(N+1)}\left(e^{\beta\eta_s} - 1 - \beta\eta_s\right)}_{\text{training-path error}} + \underbrace{\frac{\eta_s\sigma_z}{\sqrt{m\rho}}}_{\text{checkpoint error}} + \underbrace{\eta_s\varepsilon_B}_{\text{gradient-estimation error}}. \tag{9}$$

For $\beta = 0$, *the first term is defined by continuity as zero. Moreover, it is at most* $G\beta\eta_s^2 e^{\beta\eta_s}/(N+1)$.

Theorem 3.2 requires smoothness, but not convexity or an inverse Hessian. Its first term is second order in the cumulative step size when $\beta\eta_s$ is small, its checkpoint term decreases as $m^{-1/2}$, and its data-insertion dependence appears explicitly through $1/(N+1)$. This structure also clarifies the limits of the guarantee: fewer checkpoints increase sampling error, while a long or unstable training trajectory increases path error. The theorem controls a projected parameter-displacement proxy, not downstream utility or ranking accuracy. It applies to the full-batch dynamics in Eq. (3); stateful optimizers such as Adam require an augmented-state analysis and are outside the present guarantee.

Relation to the converged target. If one instead defines $\mathcal{I}^{\Pi}_{p,*}(z) = \|\Pi(\theta^*_{\mathcal{D}+z} - \theta^*_{\mathcal{D}})\|_2$, then the right-hand side of Eq. (9) must additionally include

$$r_{\mathcal{D}}(T) + r_{\mathcal{D}+z}(T), \quad r_{\mathcal{A}}(T) = \|\Pi(\theta^T_{\mathcal{A}} - \theta^*_{\mathcal{A}})\|_2. \tag{10}$$

These optimization residuals cannot be removed without assumptions that guarantee convergence. We therefore use the finite-horizon score in Eq. (4), which matches the fixed-budget retraining protocol used for evaluation.

3.2.2 Complexity

Computing Eq. (6) over all T checkpoints and a scored parameter block of dimension p_Π costs $\mathcal{O}(Tp_\Pi)$ per candidate. The Monte Carlo estimator in Eq. (7) reduces this to $\mathcal{O}(mp_\Pi)$. We use $m = 5$ checkpoints and restrict Π to the LM head. The training-data gradients are cached during surrogate training, and mini-batch estimates replace full-data gradients in practice. Their approximation error is represented explicitly by ε_B in Theorem 3.2.

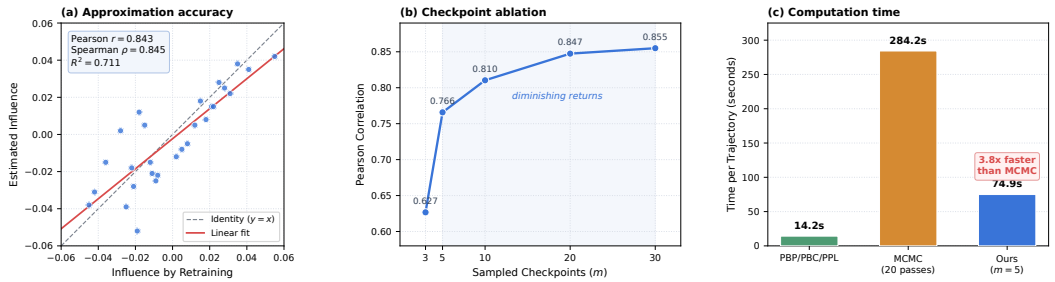


Figure 4: **Accuracy and efficiency of Pseudo-Influence on synthetic TERMINAL-BENCH trajectories using Qwen3-32B.** (a) Estimated influence versus influence obtained by full retraining across 28 trajectories; the dashed line denotes $y = x$ and the red line is a linear fit. (b) Pearson correlation with retraining-based influence as the number of sampled checkpoints m increases. (c) Wall-clock scoring time per trajectory. Restricting gradients to the LM head and caching training-data gradients makes our estimator $3.8\times$ faster than 20-pass MCMC-Dropout.

3.3 ACTIVE LEARNING WITH PSEUDO-INFLUENCE

Algorithm 1 applies the estimator at each acquisition round. The surrogate first generates a pseudo-labeled rollout for every remaining task. We then evaluate Eq. (7) using the saved checkpoints and rank candidates by the resulting finite-horizon parameter influence. The highest-scoring tasks are sent to the annotation oracle, and their verified trajectories are added to $\mathcal{D}_l$. Because the score requires neither an indicative validation set nor candidate-wise retraining, the same procedure applies to command-line and tool-interaction trajectories.

3.4 DISCUSSION

We evaluate the fidelity and computational efficiency of Pseudo-Influence on synthetic trajectories constructed from TERMINAL-BENCH, using Qwen3-32B as the base model. Figure 4 summarizes the results.

3.4.1 APPROXIMATION ACCURACY

We compare the estimated influence of 28 trajectories with their influence obtained by full retraining. As shown in Figure 4(a), the estimator agrees strongly with retraining: Pearson $r = 0.843$, Spearman $\rho = 0.845$, and linear $R^2 = 0.711$. The agreement in both value and ranking indicates that the efficient estimator preserves the relative informativeness of trajectories without requiring separate retraining for every candidate.

Figure 4(b) varies the number of sampled training checkpoints m. Pearson correlation increases from 0.627 at $m = 3$ to 0.766 at $m = 5$, 0.810 at $m = 10$, 0.847 at $m = 20$, and 0.855 at $m = 30$. The marginal gain decreases from 0.139 between $m = 3$ and $m = 5$ to only 0.008 between $m = 20$ and $m = 30$. We therefore use $m = 5$ as the default, which captures most of the attainable agreement while limiting computation.

3.4.2 COMPUTATIONAL EFFICIENCY

We reduce scoring cost in two ways: gradients on the training data are cached during training, and influence estimation is restricted to the LM-head parameters rather than the full transformer. Figure 4(c) reports wall-clock time per trajectory. PBP, PBC, and PPL each require one 14.2-second model pass. MCMC-Dropout with 20 stochastic passes requires 284.2 seconds. Our estimator with $m = 5$ requires 74.9 seconds. It is therefore $3.8\times$ faster than MCMC-Dropout, while attaining a Pearson correlation of 0.766 with the retraining-based influence at the same checkpoint setting.

4 EXPERIMENTS

We evaluate Active Agentic Learning in two complementary settings: command-line environment manipulation on TERMINAL-BENCH 2.0 and tool use on ACEBENCH-AGENT. Across both settings, we compare random selection, perplexity (PPL), bits per character (BPC), MCMC-Dropout, and Pseudo-Influence (PI-Score). We additionally combine MCMC and PI-Score with a common diversity module to isolate whether Pseudo-Influence remains complementary to coverage-oriented selection.

4.1 EXPERIMENTAL SETUP

Models and active-learning protocol. We use Qwen3-8B and Qwen3-32B in thinking mode (Yang et al., 2025). Before active learning, each model receives an auxiliary warm-up stage on agent trajectories outside the target active pool. We then run five equal-budget acquisition rounds. In each round, a selection method ranks the remaining pool, the selected trajectories are added to the labeled set, and the model is fine-tuned on the updated set. The fifth round uses the full active pool; consequently, all selection methods share the same final-round result. This endpoint serves as a full-data reference, whereas earlier rounds measure annotation efficiency.

Selection baselines. Random samples uniformly from the remaining pool. PPL and BPC rank trajectories by sequence likelihood, with BPC normalizing by character count. MCMC uses 20 stochastic forward passes to estimate predictive uncertainty. For the diversity variants, we adopt D2 Pruning (Maharana et al., 2024), which constructs a neighborhood graph over sample representations and uses message passing to balance difficult examples against redundant local coverage. We use the primary acquisition score (MCMC or PI-Score) as the difficulty signal. To represent each task, we extract the base model's hidden state at the EOS token of the problem statement and use it to build the neighborhood graph. This EOS representation is our implementation choice rather than a component prescribed by D2 Pruning.

4.2 COMMAND-LINE INTERACTION

Data and evaluation. The active pool contains approximately 1.6K trajectories from CLI-Gym[2] (Lin et al., 2026), which generates scalable command-line tasks through agentic environment inversion. Because the base models are not initially optimized for agentic coding, we first warm-start them on 48K open-source software-engineering trajectories derived from SWE-family resources (Jimenez et al., 2024). This auxiliary collection is disjoint from TERMINAL-BENCH. Active learning is then performed exclusively over the CLI-Gym pool.

We evaluate on the 89 environment-manipulation and system-level tasks in TERMINAL-BENCH 2.0 (Merrill et al., 2026). Each task requires an agent to interact with a Linux environment through command-line interfaces and diagnose or resolve a realistic software or system issue. Following the benchmark protocol, a task is successful when the resulting environment passes the official verification script. We report pass@1, defined here as the percentage of tasks solved in a single run. Before target-pool active learning, Qwen3-8B and Qwen3-32B obtain pass@1 scores of 2.2 and 5.2, respectively.

Results. Figures 5(a–b) show that Pseudo-Influence concentrates useful CLI trajectories substantially earlier than uncertainty-based selection. At the first round, PI-Score+Diversity reaches 10.2 on Qwen3-8B and 17.4 on Qwen3-32B, outperforming random selection by 4.8 and 5.5 points, respectively. The advantage persists through the middle rounds: on Qwen3-8B, PI-Score+Diversity reaches 15.2 at round 3 versus 11.2 for random selection; on Qwen3-32B, it reaches 21.2 versus 18.3. At round 4, PI-Score obtains the strongest Qwen3-32B result of 22.1, exceeding both random selection (19.2) and the full-pool endpoint (20.5). The diversity module generally improves early-round selection, while the Qwen3-32B round-4 result also shows that the best balance between influence and coverage can depend on the acquisition budget.

[2] https://huggingface.co/datasets/LiberCoders/CLI-Gym

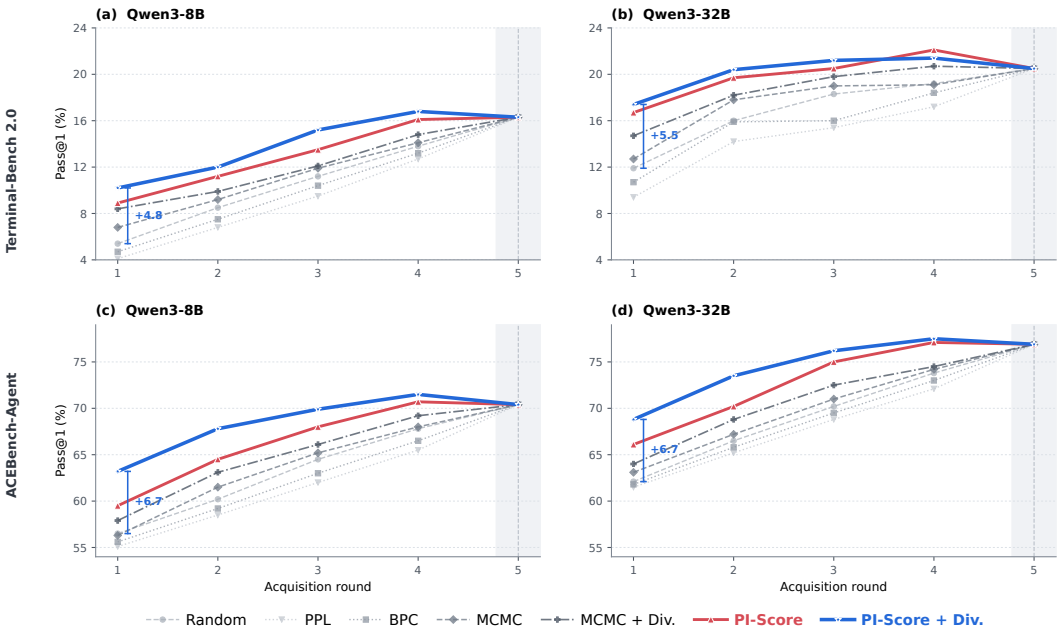


Figure 5: **Five-round active-learning performance in two agentic settings.** Pass@1 on (a–b) TERMINAL-BENCH 2.0 and (c–d) ACEBENCH-AGENT. PI-Score variants lead before full-pool training; all methods coincide at round 5. Blue brackets mark the round-1 gain over random, shaded bands denote the full-pool endpoint, and panels in each benchmark row share the same vertical range.

4.3 TOOL USE

Data and evaluation. The active pool contains approximately 4.7K executable, stateful tool-use scenarios from EnvScaler-SFT-Scenario[3] (Song et al., 2026). After warm-starting on the same 48K auxiliary trajectories, we run five active-learning rounds over this pool.

We evaluate on the agent split of ACEBench (Chen et al., 2025). We adapt its official evaluator from textual tool calls to each model's native function-calling interface and count a task as successful only when all evaluation conditions are satisfied. We report single-run pass@1; the initial Qwen3-8B and Qwen3-32B scores are 54.2 and 60.5.

Results. PI-Score+Diversity reaches 63.2 and 68.8 at round 1, improving over random by 6.7 points at both model scales. Its gains over random are 7.6 and 7.0 points at round 2 and remain 3.7 points at round 4. Thus, Pseudo-Influence transfers beyond command-line coding, with D2 diversity providing complementary gains under limited annotation budgets.

5 CONCLUSION

We introduce Active Agentic Learning and Pseudo-Influence, which estimates how a pseudo-labeled trajectory would affect training without requiring convexity or inverse-Hessian computation. Across two benchmarks and two model scales, it consistently improves pre-full-pool pass@1 over uncertainty baselines, while D2 diversity yields further gains. These results show that training impact and distributional coverage are complementary signals for acquiring agent trajectories.

REFERENCES

Dana Angluin. Queries and concept learning. *Machine Learning*, 2(4):319–342, 1988.

[3] https://huggingface.co/datasets/XXHStudyHard/EnvScaler-SFT-Scenario

David Arthur and Sergei Vassilvitskii. K-means++ the advantages of careful seeding. In *Proceedings of the eighteenth annual ACM-SIAM symposium on Discrete algorithms*, pp. 1027–1035, 2007.

Jordan T. Ash, Chicheng Zhang, Akshay Krishnamurthy, John Langford, and Alekh Agarwal. Deep batch active learning by diverse, uncertain gradient lower bounds. In *International Conference on Learning Representations*, 2020.

Les E. Atlas, David A. Cohn, and Richard E. Ladner. Training connectionist networks with queries and selective sampling. In *Advances in Neural Information Processing Systems 2*, pp. 566–573, 1989.

Samyadeep Basu, Xuchen You, and Soheil Feizi. On second-order group influence functions for black-box predictions. In *International Conference on Machine Learning*, 2020.

Zalán Bodó, Zsolt Minier, and Lehel Csató. Active learning with clustering. In *Active Learning and Experimental Design workshop In conjunction with AISTATS 2010*, pp. 127–139. JMLR Workshop and Conference Proceedings, 2011.

Daniil A. Boiko, Robert MacKnight, Ben Kline, and Gabe Gomes. Autonomous chemical research with large language models. *Nature*, 624(7992):570–578, 2023. doi: 10.1038/s41586-023-06792-0.

Olivier Borkowski, Mathilde Koch, Agnès Zettor, Amir Pandi, Angelo Cardoso Batista, Paul Soudier, and Jean-Loup Faulon. Large scale active-learning-guided exploration for in vitro protein production optimization. *Nature communications*, 11(1):1872, 2020.

Chen Chen, Xinlong Hao, Weiwen Liu, Xu Huang, Xingshan Zeng, Shuai Yu, Dexun Li, Yuefeng Huang, Xiangcheng Liu, Xinzhi Wang, and Wu Liu. Acebench: A comprehensive evaluation of llm tool usage. In *Findings of the Association for Computational Linguistics: EMNLP 2025*, pp. 12970–12998, 2025. doi: 10.18653/v1/2025.findings-emnlp.697. URL https://aclanthology.org/2025.findings-emnlp.697/.

Zehui Chen, Kuikun Liu, Qiuchen Wang, Wenwei Zhang, Jiangning Liu, Dahua Lin, Kai Chen, and Feng Zhao. Agent-FLAN: Designing data and methods of effective agent tuning for large language models. In *Findings of the Association for Computational Linguistics: ACL 2024*, pp. 9354–9366, 2024. doi: 10.18653/v1/2024.findings-acl.557.

Cody Coleman, Christopher Yeh, Stephen Mussmann, Baharan Mirzasoleiman, Peter Bailis, Percy Liang, Jure Leskovec, and Matei Zaharia. Selection via proxy: Efficient data selection for deep learning. In *International Conference on Learning Representations*, 2019.

David Cohn, Les Atlas, and Richard Ladner. Improving generalization with active learning. *Machine Learning*, 15(2):201–221, 1994.

Zheng Dai and David K Gifford. Training data attribution for diffusion models. *arXiv preprint arXiv:2306.02174*, 2023.

Ehsan Elhamifar, Guillermo Sapiro, Allen Yang, and S Shankar Sasrty. A convex optimization framework for active learning. In *Proceedings of the IEEE International Conference on Computer Vision*, pp. 209–216, 2013.

Vitaly Feldman and Zhang Chiyuan. Scalability vs. utility: Do we have to sacrifice one for the other in data importance quantification? In *Advances in neural information processing systems*, 2020.

Alexander Freytag, Erik Rodner, and Joachim Denzler. Selecting influential examples: Active learning with expected model output changes. In *European Conference on Computer Vision*, pp. 562–577, 2014.

Yarin Gal and Zoubin Ghahramani. Dropout as a bayesian approximation: Representing model uncertainty in deep learning. In *International Conference on Machine Learning*, pp. 1050–1059, 2016.

Yarin Gal, Riashat Islam, and Zoubin Ghahramani. Deep bayesian active learning with image data. In *International Conference on Machine Learning*, pp. 1183–1192, 2017.

Yuhong Guo. Active instance sampling via matrix partition. In *NIPS*, pp. 802–810, 2010.

Zayd Hammoudeh and Daniel Lowd. Training data influence analysis and estimation: A survey. *arXiv preprint arXiv:2212.04612*, 2022.

Neil Houlsby, Ferenc Huszár, Zoubin Ghahramani, and Máté Lengyel. Bayesian active learning for classification and preference learning. In *arXiv preprint arXiv:1112.5745*, 2011.

Shengding Hu, Zheng Xiong, Meng Qu, Xingdi Yuan, Marc-Alexandre Côté, Zhiyuan Liu, and Jian Tang. Graph policy network for transferable active learning on graphs. *Advances in Neural Information Processing Systems*, 33:10174–10185, 2020.

Ruoxi Jia, Fan Wu, Xuehui Sun, Jiacen Xu, David Dao, Bhavya Kailkhura, Ce Zhang, Bo Li, and Dawn Song. Scalability vs. utility: Do we have to sacrifice one for the other in data importance quantification? In *IEEE / CVF Computer Vision and Pattern Recognition Conference*, 2021.

Carlos E. Jimenez, John Yang, Alexander Wettig, Shunyu Yao, Kexin Pei, Ofir Press, and Karthik Narasimhan. SWE-bench: Can language models resolve real-world GitHub issues? In *International Conference on Learning Representations*, 2024.

Ajay J Joshi, Fatih Porikli, and Nikolaos Papanikolopoulos. Multi-class active learning for image classification. In *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2372–2379, 2009.

Pang Koh, Kai-Siang Ang, Hubert H. K. Teo, and Percy Liang. On the accuracy of influence functions for measuring group effects. In *Advances in neural information processing systems*, 2019.

Pang Wei Koh and Percy Liang. Understanding black-box predictions via influence functions. In *International Conference on Machine Learning*, 2017.

Shuming Kong, Yanyan Shen, and Linpeng Huang. Resolving training biases via influence-based data relabeling. In *International Conference on Learning Representations*, 2022.

Harrison Lee, Samrat Phatale, Hassan Mansoor, Thomas Mesnard, Johan Ferret, Kellie Ren Lu, Colton Bishop, Ethan Hall, Victor Carbune, Abhinav Rastogi, and Sushant Prakash. RLAIF vs. RLHF: Scaling reinforcement learning from human feedback with ai feedback. In *International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, 2024.

David D. Lewis and William A. Gale. A sequential algorithm for training text classifiers. In *SIGIR '94: Proceedings of the 17th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 3–12, 1994.

Yusong Lin, Haiyang Wang, Shuzhe Wu, Lue Fan, Feiyang Pan, Sanyuan Zhao, and Dandan Tu. Cli-gym: Scalable cli task generation via agentic environment inversion. *arXiv preprint arXiv:2602.10999*, 2026. URL https://arxiv.org/abs/2602.10999.

Xiao Liu, Hao Yu, Hanchen Zhang, Yifan Xu, Xuanyu Lei, Hanyu Lai, Yu Gu, Hangliang Ding, Kaiwen Men, Kejuan Yang, Shudan Zhang, Xiang Deng, Aohan Zeng, Zhengxiao Du, Chenhui Zhang, Sheng Shen, Tianjun Zhang, Yu Su, Huan Sun, Minlie Huang, Yuxiao Dong, and Jie Tang. Agentbench: Evaluating LLMs as agents. In *International Conference on Learning Representations*, 2024.

Zhuoming Liu, Hao Ding, Huaping Zhong, Weijia Li, Jifeng Dai, and Conghui He. Influence selection for active learning. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 9274–9283, 2021.

Oisin Mac Aodha, Neill DF Campbell, Jan Kautz, and Gabriel J Brostow. Hierarchical subquery evaluation for active learning on a graph. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 2651–2658, 2011.

Adyasha Maharana, Prateek Yadav, and Mohit Bansal. D2 pruning: Message passing for balancing diversity & difficulty in data pruning. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=thbtoAkCe9.

Mansheej Paul, Surya Ganguli, and Gintare Karolina Dziugaite. Deep learning on a data diet: Finding important examples early in training. In *Advances in Neural Information Processing Systems*, 2018.

Mike A. Merrill, Alexander Glenn Shaw, Nicholas Carlini, Boxuan Li, Harsh Raj, Ivan Bercovich, Lin Shi, Jeong Yeon Shin, Thomas Walshe, E. Kelly Buchanan, Junhong Shen, Guanghao Ye, Haowei Lin, Jason Poulos, Maoyu Wang, Jenia Jitsev, Marianna Nezhurina, Di Lu, Orfeas Menis Mastromichalakis, Zhiwei Xu, Zizhao Chen, Yue Liu, Robert Zhang, Leon Liangyu Chen, Anurag Kashyap, Jan-Lucas Uslu, Jeffrey Li, Jianbo Wu, Minghao Yan, Song Bian, Vedang Sharma, Ke Sun, Steven Dillmann, Akshay Anand, Andrew Lanpouthakoun, Bardia Koopah, Changran Hu, Etash Kumar Guha, Gabriel H. S. Dreiman, Jiacheng Zhu, Karl Krauth, Li Zhong, Niklas Muennighoff, Robert Kwesi Amanfu, Shangyin Tan, Shreyas Pimpalgaonkar, Tushar Aggarwal, Xiangning Lin, Xin Lan, Xuandong Zhao, Yiqing Liang, Yuanli Wang, Zilong Wang, Changzhi Zhou, David Heineman, Hange Liu, Harsh Trivedi, John Yang, Junhong Lin, Manish Shetty, Michael Yang, Nabil Omi, Negin Raoof, Shanda Li, Terry Yue Zhuo, Wuwei Lin, Yiwei Dai, Yuxin Wang, Wenhao Chai, Shang Zhou, Dariush Wahdany, Ziyu She, Jiaming Hu, Zhikang Dong, Yuxuan Zhu, Sasha Cui, Ahson Saiyed, Arinbjörn Kolbeinsson, Christopher Michael Rytting, Ryan Marten, Yixin Wang, Alex Dimakis, Andy Konwinski, and Ludwig Schmidt. Terminal-bench: Benchmarking agents on hard, realistic tasks in command line interfaces. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=a7Qa4CcHak.

Hieu T Nguyen and Arnold Smeulders. Active learning using pre-clustering. In *Proceedings of the twenty-first international conference on Machine learning*, 2004.

OpenAI and et.al. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023.

Ozan Sener and Silvio Savarese. Active learning for convolutional neural networks: A coreset approach. In *International Conference on Learning Representations*, 2018.

Sung Min Park, Kristian Georgiev, Andrew Ilyas, Guillaume Leclerc, and Aleksander Madry. Trak: Attributing model behavior at scale. *arXiv preprint arXiv:2303.14186*, 2023.

Garima Pruthi, Frederick Liu, Sundararajan Mukund, and Satyen Kale. Estimating training data influence by tracing gradient descent. *arXiv preprint arXiv:2002.08484*, 2020.

Dan Roth and Kevin Small. Margin-based active learning for structured output spaces. In *European Conference on Machine Learning*, pp. 413–424, 2006.

Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, and others. Imagenet large scale visual recognition challenge. *International journal of computer vision*, 115(3):211–252, 2015.

Timo Schick, Jane Dwivedi-Yu, Roberto Dessi, Roberta Raileanu, Maria Lomeli, Eric Hambro, Luke Zettlemoyer, Nicola Cancedda, and Thomas Scialom. Toolformer: Language models can teach themselves to use tools. In *Advances in Neural Information Processing Systems*, 2023. doi: 10.52202/075280-2997.

Burr Settles and Mark Craven. An analysis of active learning strategies for sequence labeling tasks. In *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing*, pp. 1070–1079, 2008.

Noah Shinn, Federico Cassano, Ashwin Gopinath, Karthik Narasimhan, and Shunyu Yao. Reflexion: Language agents with verbal reinforcement learning. In *Advances in Neural Information Processing Systems*, 2023. doi: 10.52202/075280-0377.

Xiaoshuai Song, Haofei Chang, Guanting Dong, Yutao Zhu, Ji-Rong Wen, and Zhicheng Dou. Envscaler: Scaling tool-interactive environments for llm agent via programmatic synthesis. In *Findings of the Association for Computational Linguistics: ACL 2026*, pp. 8326–8357, 2026. doi: 10.18653/v1/2026.findings-acl.407. URL https://aclanthology.org/2026.findings-acl.407/.

Dan Wang and Yi Shang. A new active labeling method for deep learning. In *International joint conference on neural networks (IJCNN)*, pp. 112–119, 2014.

Keze Wang, Dongyu Zhang, Ya Li, Ruimao Zhang, and Liang Lin. Cost-effective active learning for deep image classification. *IEEE Transactions on Circuits and Systems for Video Technology*, 27 (12):2591–2600, 2016.

Lei Wang, Chen Ma, Xueyang Feng, Zeyu Zhang, Hao Yang, Jingsen Zhang, Zhiyuan Chen, Jiakai Tang, Xu Chen, Yankai Lin, Wayne Xin Zhao, Zhewei Wei, and Ji-Rong Wen. A survey on large language model based autonomous agents. *arXiv preprint arXiv:2308.11432*, 2023. doi: 10.1007/s11704-024-40231-1.

Tianyang Wang, Xingjian Li, Pengkun Yang, Guosheng Hu, Xiangrui Zeng, Siyu Huang, Cheng-Zhong Xu, and Min Xu. Boosting active learning via improving test performance. *arXiv preprint arXiv:2112.05683*, 2021.

Manfred KK Warmuth, Gunnar Rätsch, Michael Mathieson, Jun Liao, and Christian Lemmen. Active learning in the drug discovery process. *Advances in Neural information processing systems*, 14, 2001.

Jiaxi Wu, Jiaxin Chen, and Di Huang. Entropy-based active learning for object detection with progressive diversity constraint. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 9397–9406, 2022.

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, Wensen Cheng, Qi Zhang, Wenjuan Qin, Yongyan Zheng, Xipeng Qiu, Xuanjing Huang, and Tao Gui. The rise and potential of large language model based agents: A survey. *arXiv preprint arXiv:2309.07864*, 2023.

Yiheng Xu, Dunjie Lu, Zhennan Shen, Junli Wang, Zekun Wang, Yuchen Mao, Caiming Xiong, and Tao Yu. Agenttrek: Agent trajectory synthesis via guiding replay with web tutorials. In *International Conference on Learning Representations*, 2025.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, et al. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*, 2025. URL https://arxiv.org/abs/2505.09388.

Shuo Yang, Zeke Xie, Hanyu Peng, Min Xu, Mingming Sun, and Ping Li. Dataset pruning: Reducing training data by examining generalization influence. In *International Conference on Learning Representations*, 2023.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. ReAct: Synergizing reasoning and acting in language models. In *International Conference on Learning Representations*, 2023.

Shunyu Yao, Noah Shinn, Pedram Razavi, and Karthik Narasimhan. τ-bench: A benchmark for tool-agent-user interaction in real-world domains. In *International Conference on Learning Representations*, 2025.

Changchang Yin, Buyue Qian, Shilei Cao, Xiaoyu Li, Jishang Wei, Qinghua Zheng, and Ian Davidson. Deep similarity-based batch mode active learning with exploration-exploitation. In *2017 IEEE International Conference on Data Mining (ICDM)*, pp. 575–584, 2017.

Donggeun Yoo and In So Kweon. Learning loss for active learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 93–102, 2019.

Tianning Yuan, Fang Wan, Mengying Fu, Jianzhuang Liu, Songcen Xu, Xiangyang Ji, and Qixiang Ye. Multiple instance active learning for object detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 5330–5339, 2021.

Xueying Zhan, Qingzhong Wang, Kuan-hao Huang, Haoyi Xiong, Dejing Dou, and Antoni B Chan. A comparative survey of deep active learning. *arXiv preprint arXiv:2203.13450*, 2022.

Wentao Zhang, Yexin Wang, Zhenbang You, Meng Cao, Ping Huang, Jiulong Shan, Zhi Yang, and Bin Cui. Rim: Reliable influence-based active learning on graphs. *Advances in Neural Information Processing Systems*, 34:27978–27990, 2021.

Fedor Zhdanov. Diverse mini-batch active learning. *arXiv preprint arXiv:1901.05954*, 2019.

Shuyan Zhou, Frank F. Xu, Hao Zhu, Xuhui Zhou, Robert Lo, Abishek Sridhar, Xianyi Cheng, Tianyue Ou, Yonatan Bisk, Daniel Fried, Uri Alon, and Graham Neubig. Webarena: A realistic web environment for building autonomous agents. In *International Conference on Learning Representations*, 2024.

Broader Impact

This paper presents a novel and effective active learning algorithm to advance the deep learning area. We believe that this approach has positive implications for understanding the impact of samples and reducing the reliance on human labor for large-scale data annotation and cleaning in society. As for negative impacts, we believe that this technology, and even the entire field of artificial intelligence, may be applied to inhumane social control, which should be taken seriously by legislative bodies worldwide.

Limitations

This article presents a theoretically grounded and effective tool for measuring the impact of samples when the quality of pseudo-labeling is low, successfully applying it in active learning. However, due to current equipment limitations, the experiments conducted thus far have been on a relatively small scale, involving fewer than 100 million data points. In future work, we plan to explore larger experimental setups to validate our findings further. This expansion will enhance the robustness of our tool and its applicability in real-world scenarios, ultimately contributing to more effective active learning strategies.

AI-Usage Statement

In the preparation of this manuscript, artificial intelligence (AI) tools were employed exclusively for language polishing, grammatical correction, and debugging of specific code segments. These tools served solely as editorial and technical assistants to enhance clarity and correctness; they were not involved in any aspect of conceptualization, hypothesis generation, experimental design, literature synthesis, or autonomous research. All core intellectual contributions, including the formulation of research questions, methodological development, data interpretation, and scholarly argumentation, are entirely the product of human authorship. The authors affirm full responsibility for the originality, accuracy, and integrity of all content presented herein.

A Detailed Related Works

A.1 Active Learning

Active learning is a technique aimed at reducing the cost of annotation, and its application value in various fields has been widely explored in the academic community, such as image recognition (Wang et al., 2016; Joshi et al., 2009), object detection (Yuan et al., 2021; Wu et al., 2022), sequence processing (Settles & Craven, 2008), graph learning (Hu et al., 2020; Zhang et al., 2021), and bio-sciences (Borkowski et al., 2020; Warmuth et al., 2001). We describe the problem of active learning (AL), which aims to select the most informative data from a large unlabeled data pool for annotation until a performance level or budget is met. We use a simple classification task as an example. We start with an initial labeled set $\mathcal{D}_l$ and a large unlabeled data pool $\mathcal{D}_u$, where the unlabeled data is much more than the labeled one. The AL procedure is iterative. In each iteration, we train a model on $\mathcal{D}_l$ and select a batch of samples D_q using the model and some criteria, such as the uncertainty score (Roth & Small, 2006; Cody Coleman et al., 2019; Gal & Ghahramani, 2016). Then, an oracle, such

as a human expert or a superior commercial AI model like GPT (OpenAI & et.al, 2023), annotates the selected data. We update $\mathcal{D}_l$ and $\mathcal{D}_u$ accordingly and retrain the model on $\mathcal{D}_l$. We repeat this until the budget is used up or the model performance is satisfactory.

Depending on the criteria for selecting data, active learning techniques can be roughly divided into two categories: uncertainty-based (Roth & Small, 2006; Wang & Shang, 2014; Yoo & Kweon, 2019; Gal et al., 2017; Gal & Ghahramani, 2016) and diversity-based (Ozan Sener & Silvio Savarese, 2018; Ash et al., 2020; Elhamifar et al., 2013; Guo, 2010). Next, we will provide a brief overview of representative works in these two categories.

Uncertainty-based methods. Uncertainty-based active learning selects data points that the model is most uncertain about, reflecting the randomness and variability in how the data is generated (Zhan et al., 2022). Different criteria can be used to measure the model's uncertainty over the data samples. For instance, the margin criterion (Roth & Small, 2006) picks samples that have similar probabilities for the top two labels, while the least confidence (Wang & Shang, 2014) criterion picks samples that have the lowest probability for the most likely label. The entropy criterion (Cody Coleman et al., 2019; Wu et al., 2022) picks samples that have high predictive entropy, meaning the model is unsure about all labels. Learning Loss (Yoo & Kweon, 2019) predicts the loss of unlabeled samples using a small module attached to the target model. Bayesian Active Learning by Disagreements (Gal et al., 2017; Houlsby et al., 2011) picks samples that provide the most information about the model parameters. MCMC-Dropout (Gal & Ghahramani, 2016) uses a stochastic variational inference method to estimate the uncertainty of the samples. Gradient norm (Wang et al., 2021; Mansheej Paul et al., 2018) guides the selection of unlabeled data based on the gradient magnitude, which can lower the test loss bound.

Diversity-based methods. Another popular practice in active learning is to select samples based on the diversity criteria (Bodó et al., 2011; Nguyen & Smeulders, 2004; Mac Aodha et al., 2011), which aims to achieve good data coverage and diversity. Some methods use clustering techniques, such as k-means, to group the data in the feature space and select representative samples from the cluster centroids. To enhance the scalability of the clustering algorithms, some methods (Ozan Sener & Silvio Savarese, 2018) operate on data batches instead of the entire dataset. The classic Coreset method (Ozan Sener & Silvio Savarese, 2018) uses a greedy k-center algorithm to choose the coreset with a small radius, while BADGE (Ash et al., 2020) uses the efficient k-means++ (Arthur & Vassilvitskii, 2007) to cluster the gradient embeddings of the current model and select a subset from each cluster. Some methods (Elhamifar et al., 2013; Guo, 2010) also use discrete optimization algorithms to solve for the optimal covering subset. CPAL (Elhamifar et al., 2013) uses convex programming to select multiple informative and diverse unlabeled samples for any classifier, while Guo (Guo, 2010) uses a Gaussian process framework to select queries by maximizing the mutual information between labeled and unlabeled data and solves the NP-hard matrix partition problem by a local optimization technique. Furthermore, some work (Zhdanov, 2019; Yin et al., 2017) employs a hybrid strategy, where these methods first use uncertainty-based metrics to select a subset of high-information samples and then, in a second stage, run diversity-based clustering selection methods on these selected samples.

Comparison of ours with other active learning methods. Based on their selection criteria, existing active learning techniques can be roughly divided into two categories. The first category is based on sample-wise uncertainty (Roth & Small, 2006; Wang & Shang, 2014; Yoo & Kweon, 2019; Gal et al., 2017; Gal & Ghahramani, 2016), where these methods rely on pre-defined uncertainty measures. However, neural networks sometimes produce abnormal confidence, *e.g.* generating wrong predictions with high certainty, which makes these uncertainty measures not always stable. On the other hand, the second category of existing methods is based on diversity (Ozan Sener & Silvio Savarese, 2018; Ash et al., 2020; Elhamifar et al., 2013; Guo, 2010), but these approaches rely on clustering in feature space, which limits their ability to scale to larger datasets. In contrast, our approach does not rely on defining uncertainty or on complex feature-space clustering. Instead, it considers the influence of samples, naturally avoiding the inherent problems of the aforementioned two categories of methods.

A.2 Agentic AI and Trajectory Learning

The deployment of large language models as autonomous agents has emerged as a major frontier in artificial intelligence (Xi et al., 2023; Wang et al., 2023). Unlike traditional question-answering

systems that map a single input to a single output, agentic systems operate through multi-step interactions with external environments: issuing tool calls, parsing structured API responses, maintaining internal state, and planning over extended horizons to accomplish complex tasks. Representative benchmarks such as WebArena (Zhou et al., 2024), τ-bench (Yao et al., 2025), SWE-bench (Jimenez et al., 2024), and AgentBench (Liu et al., 2024) evaluate agents across diverse domains including web navigation, software engineering, database querying, and customer service.

Training effective agents requires high-quality trajectory annotations, where each trajectory consists of interleaved observations, reasoning steps, and tool invocations. Several paradigms have been developed to produce such data. Yao et al. (2023) propose the ReAct framework, which interleaves chain-of-thought reasoning with action execution, providing a structured template for trajectory construction. Schick et al. (2023) demonstrate that language models can learn to invoke tools through self-supervised trajectory annotation. More recently, AgentTrek (Xu et al., 2025) synthesizes agent trajectories by replaying web tutorials, while Agent-FLAN (Chen et al., 2024) systematically designs training data and fine-tuning strategies for effective agent tuning. Self-improvement approaches such as Reflexion (Shinn et al., 2023) enable agents to iteratively refine their policies through verbal self-reflection, and RLAIF (Lee et al., 2024) scales reinforcement learning by using AI-generated feedback as a substitute for costly human annotations.

Despite these advances, a critical bottleneck remains: the annotation cost of agentic trajectories is orders of magnitude higher than that of standard QA pairs. Each trajectory may span hundreds of tool-calling steps, requires domain expertise to verify correctness, and involves non-trivial environment interactions that are difficult to automate. This high annotation cost motivates the application of active learning to agentic data collection, a direction that, to the best of our knowledge, has not been systematically explored.

A.3 Influence Analysis

Influence analysis (Koh & Liang, 2017; Pruthi et al., 2020; Park et al., 2023; Dai & Gifford, 2023; Koh et al., 2019; Yang et al., 2023) aims to measure the significance of a sample on the training results. One way to calculate the influence or contribution of a single sample to the learning process is by computing its leaving-one-out (LOO) retraining error. Formally, it is,

$$\mathcal{I}_z = \mathcal{L}(\mathcal{V}, \theta^*_{\mathcal{D}/z}) - \mathcal{L}(\mathcal{V}, \theta^*_{\mathcal{D}}), \tag{11}$$

where $\theta^*_{\mathcal{D}/z}$ and $\theta^*_{\mathcal{D}}$ indicate the optimal parameter optimized on $\mathcal{D}/z$ (the training set exculding z) and $\mathcal{D}$, and $\mathcal{L}(\mathcal{V}, \theta) = \sum_{z \in \mathcal{V}} \ell(z, \theta)$ measures the loss on the indicative set $\mathcal{V}$ of the network parameterized by θ. It elucidates the connection between training data and model predictions.

However, the high computational expense of retraining for each sample is generally prohibitive due to the enormous size of contemporary deep-learning datasets (Russakovsky et al., 2015). Even some works that try to reduce the cost of retraining, such as by utilizing the KNN classifier (Jia et al., 2021) and dividing the full set into subsets (Feldman & Chiyuan, 2020), still incur a high overall expense. Instead of performing any retraining, Koh & Liang (2017) provides us with a closed-form and affordable approximator. By supposing the empirical risk is second-order differentiable and strictly convex in θ, it provides a computable estimator for the influence on the empirical loss,

$$\mathcal{I}_z = -\nabla \mathcal{L}(\mathcal{V}, \theta^*_{\mathcal{D}})^{\mathrm{T}} H^{-1} \nabla \ell(z, \theta^*_{\mathcal{D}}), \tag{12}$$

where $H = \nabla^2 \mathcal{L}(\mathcal{V}, \theta^*_{\mathcal{D}})$ is the Hessian of the converged parameters and is assumed to be positive definite. The proposal of influence functions has motivated the community to explore related applications (Liu et al., 2021; Kong et al., 2022) and further improvements (Koh et al., 2019; Basu et al., 2020; Pruthi et al., 2020). Some existing works (Liu et al., 2021; Freytag et al., 2014) on active learning design influence measurement and use it to select data points that have a higher impact on the training process. However, the vanilla influence function has several limitations, such as assuming a strongly convex loss function and using only the parameters after strict convergence, which ignores the training dynamics (Hammoudeh & Lowd, 2022).

B Proofs for the Pseudo-Influence Approximation

B.1 Derivation of the frozen-path estimator

By the definition of the mean empirical objective in Eq. (2), at every base checkpoint $\theta_{\mathcal{D}}^t$ we have the exact identity

$$\begin{aligned}\nabla F_{\mathcal{D}+z}(\theta_{\mathcal{D}}^t) - \nabla F_{\mathcal{D}}(\theta_{\mathcal{D}}^t) &= \frac{NG_{\mathcal{D}}^t + G_z^t}{N+1} - G_{\mathcal{D}}^t \\ &= \frac{G_z^t - G_{\mathcal{D}}^t}{N+1} = h_z^t.\end{aligned} \tag{13}$$

Thus, unlike the path approximation below, the factor $1/(N+1)$ and the gradient difference in Eq. (5) follow exactly from inserting one sample into a mean empirical risk.

Proof of Proposition 1. Subtracting the two updates in Eq. (3) gives

$$\begin{aligned}\Delta_z^{t+1} &= \Delta_z^t - \eta_t \left[\nabla F_{\mathcal{D}+z}(\theta_{\mathcal{D}+z}^t) - \nabla F_{\mathcal{D}}(\theta_{\mathcal{D}}^t)\right] \\ &= \Delta_z^t - \eta_t h_z^t - \eta_t a_z^t,\end{aligned} \tag{14}$$

where a_z^t is defined in Proposition 1. Both trajectories use the same initialization, so $\Delta_z^0 = 0$. Summing Eq. (14) over $t = 0, \ldots, T-1$ and applying Π yields

$$\Pi\Delta_z^T = -\sum_{t=0}^{T-1} \eta_t \Pi h_z^t - \sum_{t=0}^{T-1} \eta_t \Pi a_z^t, \tag{15}$$

which is Eq. (8). Dropping the second sum gives $\bar{\Delta}_T(z)$. Finally, because $P(t) = \eta_t/\eta_s$, $\sum_t \eta_t \Pi h_z^t = \eta_s \mathbb{E}_{t\sim P}[\Pi h_z^t]$, proving Eq. (6). □

B.2 Proof of the finite-horizon bound

Proof of Theorem 3.2. We first bound the training-path error. Smoothness and $\|\Pi\|_{\mathrm{op}} \le 1$ imply

$$\|a_z^t\|_2 \le \beta \|\Delta_z^t\|_2, \qquad \|h_z^t\|_2 \le \frac{2G}{N+1} =: H. \tag{16}$$

Let $S_t = \sum_{k=0}^{t-1} \eta_k$. From Eq. (14),

$$\|\Delta_z^{t+1}\|_2 \le (1 + \beta\eta_t)\|\Delta_z^t\|_2 + H\eta_t. \tag{17}$$

Since $\Delta_z^0 = 0$, the discrete Gronwall inequality and $1 + x \le e^x$ give

$$\|\Delta_z^t\|_2 \le \frac{H}{\beta}\left(e^{\beta S_t} - 1\right). \tag{18}$$

Using Proposition 1, Eq. (18), and the fact that $u \mapsto e^{\beta u} - 1$ is nondecreasing,

$$\|\Pi\Delta_z^T - \bar{\Delta}_T(z)\|_2 \le \beta \sum_{t=0}^{T-1} \eta_t \|\Delta_z^t\|_2 \tag{19}$$

$$\le H \sum_{t=0}^{T-1} \eta_t \left(e^{\beta S_t} - 1\right) \tag{20}$$

$$\le H \int_0^{\eta_s} \left(e^{\beta u} - 1\right) du \tag{21}$$

$$= \frac{2G}{\beta(N+1)}\left(e^{\beta\eta_s} - 1 - \beta\eta_s\right). \tag{22}$$

The limit is zero when $\beta = 0$. The inequality $e^x - 1 - x \le x^2 e^x/2$ for $x \ge 0$ also yields the simpler upper bound $G\beta\eta_s^2 e^{\beta\eta_s}/(N+1)$.

We next bound checkpoint sampling. Let $X_j = \Pi h_z^{t_j}$ and $\mu_z = \mathbb{E}[X_j]$. Independence gives

$$\mathbb{E}\left\| \frac{1}{m}\sum_{j=1}^{m} X_j - \mu_z \right\|_2^2 = \frac{\sigma_z^2}{m}. \tag{23}$$

Applying Markov's inequality to the squared norm shows that, with probability at least $1-\rho$,

$$\left\| \frac{1}{m}\sum_{j=1}^{m} X_j - \mu_z \right\|_2 \le \frac{\sigma_z}{\sqrt{m\rho}}. \tag{24}$$

The assumed mini-batch error further gives

$$\left\| \frac{1}{m}\sum_{j=1}^{m} \Pi(\widetilde{h}_z^{t_j} - h_z^{t_j}) \right\|_2 \le \varepsilon_B. \tag{25}$$

Finally, the reverse triangle inequality implies

$$\left| \|\Pi\Delta_z^T\|_2 - \left\| -\frac{\eta_s}{m}\sum_{j=1}^{m} \Pi\widetilde{h}_z^{t_j} \right\|_2 \right| \le \left\| \Pi\Delta_z^T + \frac{\eta_s}{m}\sum_{j=1}^{m} \Pi\widetilde{h}_z^{t_j} \right\|_2. \tag{26}$$

Combining Eqs. (22)–(25) with the triangle inequality proves Eq. (9). □

B.3 Why the converged target needs optimization residuals

Let $\mathcal{I}_{p,*}^{\Pi}(z) = \|\Pi(\theta^*_{\mathcal{D}+z} - \theta^*_{\mathcal{D}})\|_2$. Another application of the reverse triangle inequality gives

$$\left|\mathcal{I}_{p,*}^{\Pi}(z) - \mathcal{I}_{p,T}^{\Pi}(z)\right| \le \|\Pi(\theta^*_{\mathcal{D}+z} - \theta^T_{\mathcal{D}+z})\|_2 + \|\Pi(\theta^*_{\mathcal{D}} - \theta^T_{\mathcal{D}})\|_2. \tag{27}$$

Therefore, a valid bound for the converged score is Eq. (9) plus $r_{\mathcal{D}+z}(T) + r_{\mathcal{D}}(T)$. Without a condition such as strong convexity or a problem-specific convergence guarantee, these residuals cannot be replaced by a vanishing explicit rate. This is why our main result is stated for the finite training budget used by the actual acquisition procedure.