YOU DON'T NEED THE WHOLE META-DATASET: BUDGETED MODEL EVALUATION ON UNSEEN AND UNLABELED DATA

Anonymous authors
Paper under double-blind review

ABSTRACT

Predicting model performance on an unseen and unlabeled workload commonly starts by measuring its distribution shift from the training data. To convert this shift into an accuracy estimate, prior works train an evaluator on a diverse meta-dataset, using the shift between each subset of the meta-dataset and the training data as input, and the model accuracy on that subset as the label. However, obtaining these labels is expensive because each model must be run and evaluated on every subset. The more subsets the meta-dataset contains, the more forward passes are required from each reference model to obtain its accuracy labels before the evaluator can be trained. To overcome this limitation, we present ActiveEvaluator, which trains the same evaluator on a small representative subset of the meta-dataset instead of the full one, reducing the cost and latency of model evaluation to nearly one third while matching or even outperforming full-set training. ActiveEvaluator measures subset similarity using Hausdorff distance and maximizes a facility-location objective that assigns each subset to its closest representative, giving redundant subsets little marginal gain. The objective is monotone and submodular, so greedy selection comes with a provable performance guarantee. We validate ActiveEvaluator across Text2SQL, image classification, and node classification, showing that the same budgeted selection principle generalizes across structurally different tasks. As data and model pools grow faster than the resources available to evaluate them, this opens a direction toward fast, budgeted evaluation of new models on unlabeled data. It also makes evaluation more practical in domains where data are limited. Our code is available at `https://anonymous.4open.science/r/ActiveEvaluator-00B1/`.

1 INTRODUCTION

Models are increasingly deployed on workloads whose scale outpaces our ability to obtain ground-truth labels. Meta (Karpathiotakis et al., 2025), for example, ingests more than 15 TB of new data every second. Labeling such workloads is costly (Yu et al., 2018; Boyeau et al., 2025), slow (Haas et al., 2015), and often constrained by privacy (Kairouz et al., 2021), so models may be deployed before their accuracy can be measured. Meanwhile, the number of available models is also growing rapidly: public Hugging Face repositories expanded from 2.43 million to 2.96 million models in the first seven months of 2026 (Yakefu et al., 2026), spanning vision (Dosovitskiy et al., 2021; Radford et al., 2021), graph learning (Kipf & Welling, 2017; Rampasek et al., 2022), and Text2SQL (Li et al., 2024; 2025). This gap between model availability and label scarcity raises a fundamental question: *How accurately will a model perform on a new dataset when no labels are available?*

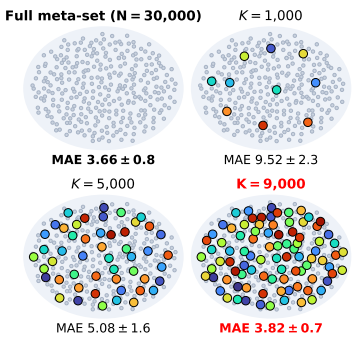

Figure 1: Hausdorff facility-location selection of K subsets from a 30K meta-set. Avg. MAE with 95% CI.

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.

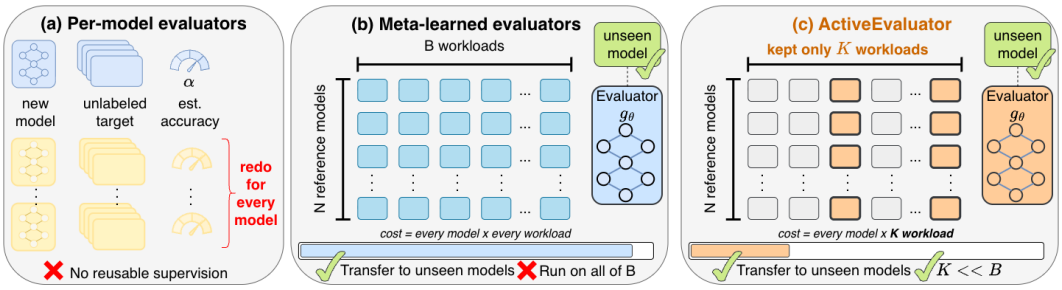

Figure 2: Representative meta-dataset selection. **(a)** Label-free estimators evaluate one model at a time. **(b)** A meta-learned evaluator trains on the full meta-dataset B. **(c)** ActiveEvaluator trains the same evaluator on a budgeted subset of only $K \ll |B|$ workloads.

The standard approach is to quantify the *distribution shift* between a model's training data and its deployment workload, then convert this magnitude into an accuracy estimate. Existing methods measure this shift in three ways: **(1)** from the model's confidence on the target, as in ATC (Garg et al., 2022), DoC (Guillory et al., 2021), and prediction dispersity (Deng et al., 2023), **(2)** from distances between training and target distributions (Deng & Zheng, 2021; Zheng et al., 2023; Pham et al., 2026b), or **(3)** with a meta-learned evaluator, MetaEvaluator (Pham et al., 2026a), which learns the relationship between distribution shift and accuracy across many reference models and transfers that relationship to unseen models.

Distribution shift is only one part of the problem. Existing evaluators require supervision to learn how the shift between training data and a target workload relates to model accuracy. We call the collection of target subsets used for this supervision the *meta-dataset*. For each subset, the evaluator uses its distribution shift as input and the model accuracy on that subset as the supervision label. Obtaining this label requires running and evaluating the model on the subset. Therefore, labeling a meta-dataset of N subsets requires N model–subset evaluations for a single model.

These evaluations are becoming increasingly expensive. Recent reasoning models improve accuracy by generating longer chains of thought (Sui et al., 2025; Chen et al., 2025), while autoregressive decoding makes attention cost grow with output length (Zhang et al., 2025; Aghajohari et al., 2026). Frontier-model inference prices have increased by $3\times$ to $18\times$ per year (Gundlach et al., 2025), and evaluating only 9 models on 9 agent benchmarks has already consumed 2.5B tokens and about $40,000 (Kapoor et al., 2026). Larger models also require more hardware and incur higher latency and energy consumption (Erdil, 2025; Kim et al., 2026).

However, once trained, a conventional evaluator learns the shift–accuracy relationship for only one model. A new model must be evaluated again on the meta-dataset to obtain its accuracy labels. MetaEvaluator (Pham et al., 2026a) addresses this limitation by meta-learning across multiple reference models, allowing knowledge of shift–accuracy relationships to transfer to unseen models. However, this broader supervision increases the cost from N to $N \times R$ model–subset evaluations for R reference models. As both the meta-dataset and the reference-model pool grow, evaluating every subset becomes increasingly expensive and slow, even though many subsets may provide redundant supervision. This raises a natural question: *Can we choose only a small set of representative subsets that retains nearly the same information as the full meta-dataset?*

To address this question, we propose **ActiveEvaluator**, which selects a representative collection of subsets from the meta-dataset before running any reference models. It employs a Hausdorff-distance-based acquisition function to identify a minimal yet representative collection, thereby reducing the evaluation budget for underrepresented workloads. Existing unlabeled evaluation pipelines are then trained on the selected collection of subsets. As shown in Fig. 1, selecting 9,000 of 30,000 subsets yields an MAE of 3.82, close to the 3.66 of the full meta-dataset, while narrowing the 95% CI from 0.8 to 0.7. We attribute the lower variance to removing redundant, noisy workloads. Our contributions are:

- **Formulation.** We formulate *budgeted evaluation* for unseen models on unseen and unlabeled data. Existing evaluators depend on costly supervision over large meta-datasets, which scales poorly as new data and models emerge.

- **Method.** We propose ActiveEvaluator, which selects a small set of subsets that best represents the full meta-dataset before supervision. It combines a Hausdorff-distance-based metric with a budgeted facility-location objective over subset embeddings. The objective is monotone submodular, giving greedy selection a $(1 - 1/e)$ approximation guarantee while substantially reducing model–subset evaluations.
- **Evaluation.** We focus on Text2SQL accuracy estimation and model ranking, with complementary image and node classification results reported in the appendix. In all settings, ActiveEvaluator matches or outperforms existing evaluators at only one third or less of the evaluation cost.
- **Benchmarking.** ActiveEvaluator supports fast, reliable benchmarking of unseen models across three modalities. At a smaller evaluation budget, it matches the ranking quality of full meta-dataset training, enabling reliable top-k model selection when exact accuracy estimates remain uncertain.

2 Related Work

Model evaluation commonly relies on benchmarks with task-specific ground truth, such as HumanEval for code generation (Chen et al., 2021), MATH for competition mathematics (Hendrycks et al., 2021b), and GSM8K for grade-school math reasoning (Cobbe et al., 2021). However, labeling new evaluation data is costly. Therefore, we focus on the unlabeled setting across three modalities: Text2SQL, image classification, and node classification. They represent relational, visual, and graph-structured data, respectively, allowing us to assess whether our method generalizes across distinct data types under a fixed evaluation budget. Even with imperfect estimates, an evaluator remains useful if it ranks candidate models correctly and identifies the top-k models for deployment.

Text2SQL. Text2SQL evaluation usually relies on labeled benchmarks (Hong et al., 2025; Liu et al., 2025a), from Spider (Yu et al., 2018) to recent enterprise and live settings such as BIRD (Li et al., 2023), BEAVER (Chen et al., 2024a), LiveSQLBench (BIRD Team, 2025), and Spider 2.0 (Lei et al., 2025). FusionSQL (Pham et al., 2026b) trains an evaluator on shifts between source training data and synthetic datasets to predict accuracy on unlabeled targets. MetaEvaluator (Pham et al., 2026a) meta-learns across models and shifts to transfer evaluation to unseen models.

Image classification. AutoEval (Deng & Zheng, 2021) trains an evaluator on transformed image datasets to learn how distribution shifts affect classifier accuracy on unlabeled data. MetaEvaluator (Pham et al., 2026a) extends this idea by transferring across models and adapting to unseen classifiers. Other methods estimate accuracy from confidence shifts (Guillory et al., 2021; Garg et al., 2022), model disagreement (Jiang et al., 2022; Baek et al., 2022), or disagreement discrepancy (Rosenfeld & Garg, 2023).

Node classification. GOOD (Gui et al., 2022) reports substantial drops in GNN node-classification accuracy under distribution shifts. GNNEvaluator (Zheng et al., 2023) generates augmented source graphs and trains an evaluator on embedding-based shifts to predict a pretrained GNN's accuracy on unseen and unlabeled graphs.

As shown in Fig. 2, across these settings, existing works focus mainly on inferring model accuracy from distribution shift. However, their supervised evaluator training is expensive because generating supervised data requires many forward passes through reference models. As a result, we study the complementary problem of constructing evaluator supervision efficiently by selecting a small, representative subset under a fixed evaluation budget.

3 Problem Formulation

First, we formulate label-free evaluation for a single model and then extend it to unseen models through meta-learning, as in MetaEvaluator. Building on this setting, we define the problem of budgeted meta-dataset construction. Throughout this section, n_j denotes the number of examples in workload j, while N denotes the number of available workloads.

Label-free evaluation. Let $f(\cdot \mid \psi)$ be a trained predictive model with parameters ψ and source training data $\mathcal{D}_S$. Let $B = \{b_1, \ldots, b_N\}$ be the meta-dataset of N labeled workloads available

for evaluator training, where $b_j = \{(x_i^j, y_i^j)\}_{i=1}^{n_j}$. Here, x_i^j is the i-th input in workload j, and y_i^j is its ground-truth label. For each workload, a label-free evaluator computes a shift descriptor $\mathrm{SD}_j = \mathrm{SD}(\mathcal{D}_S, b_j, f)$ and pairs it with the true performance of f, $a_j = \frac{1}{n_j}\sum_{i=1}^{n_j} \mu\Big(f(x_i^j \mid \psi), y_i^j\Big)$, where $\mu(\cdot,\cdot)$ is the task metric, such as execution accuracy or top-1 accuracy. These pairs form the model-specific evaluator training set $\mathcal{E}^{(f)} = \{(\mathrm{SD}_j, a_j)\}_{j=1}^{N}$.

The evaluator $g_\theta^{(f)}$ is trained by minimizing $\theta^\star = \arg\min_\theta \sum_{j=1}^{N} \mathcal{L}\Big(g_\theta^{(f)}(\mathrm{SD}_j), a_j\Big)$. At deployment, the target workload $\mathcal{D}_T^{\mathrm{new}} = \{x_i^{\mathrm{new}}\}_{i=1}^{n}$ with n samples is unlabeled. Given f, the evaluator computes $\mathrm{SD}_{\mathrm{new}} = \mathrm{SD}(\mathcal{D}_S, \mathcal{D}_T^{\mathrm{new}}, f)$ and predicts $\hat{a}_{\mathrm{new}} = g_{\theta^\star}^{(f)}(\mathrm{SD}_{\mathrm{new}})$.

Unseen models. The evaluator training set $\mathcal{E}^{(f)}$ depends on the evaluated model f. Therefore, a model-specific evaluator $g_{\theta^\star}^{(f)}$ does not transfer directly to a new model. Let $\mathcal{M}$ be a pool of R models. A naive extension evaluates each $m \in \mathcal{M}$ on B and constructs:

$$\mathcal{E}^{(m)} = \{(\mathrm{SD}_j^{(m)}, a_j^{(m)})\}_{j=1}^{N}.$$

Here, $\mathrm{SD}_j^{(m)}$ describes the shift between the training data of m and workload b_j, while $a_j^{(m)}$ is the performance of m on b_j. Then, the evaluator is retrained or fine-tuned on all observed model–workload pairs:

$$\theta^\star = \arg\min_\theta \sum_{m\in\mathcal{M}} \sum_{j=1}^{N} \mathcal{L}\Big(g_\theta(\mathrm{SD}_j^{(m)}), a_j^{(m)}\Big)\,.$$

This extension has two limitations. Each newly observed model must be run and scored on B to obtain $a_j^{(m)}$. Moreover, all models share one parameter vector $\theta^\star$, which provides no model-specific adaptation for an unseen model.

Meta-learning. MetaEvaluator instead treats each $\mathcal{E}^{(m)}$ as a separate meta-learning task, making $\mathcal{M}$ the reference-model pool. These tasks form the meta-training set:

$$\mathcal{S} = \{\mathcal{E}^{(m)} : m \in \mathcal{M}\}.$$

MetaEvaluator learns shared parameters from $\mathcal{S}$ and adapts a model-specific context for each unseen model. At deployment, an unseen model $m_{\mathrm{new}} \notin \mathcal{M}$, written as $f_{\mathrm{new}}(\cdot \mid \psi_{\mathrm{new}})$ with parameters ψ_{new}, is evaluated on the previously defined unlabeled workload $\mathcal{D}_T^{\mathrm{new}}$. The model produces $\hat{y}_i = f_{\mathrm{new}}(x_i^{\mathrm{new}} \mid \psi_{\mathrm{new}})$. Its true dataset-level performance would be:

$$M^\star = \frac{1}{n}\sum_{i=1}^{n} \mu(\hat{y}_i, y_i^{\mathrm{new}\star})\,,$$

where $y_i^{\mathrm{new}\star}$ is the unknown label. Let $\widehat{M}$ denote the evaluator's estimate of $M^\star$. Its estimation error is $\mathrm{MAE} = |\widehat{M} - M^\star|$, measured over unseen models and target workloads.

Budgeted selection. Constructing $\mathcal{E}^{(f)}$ requires running model f on all N workloads in B. MetaEvaluator repeats this process for R reference models, requiring $R \times N$ model–workload runs to construct $\mathcal{S}$. However, similar workloads may provide redundant supervision. Therefore, given a budget K, we select a representative subset:

$$A \subseteq B, \qquad |A| \le K.$$

The selection is independent of the evaluator architecture. For a model-specific evaluator, such as FusionSQL or GNNEvaluator, A refers to the reduced training set:

$$\mathcal{E}_A^{(f)} = \{(\mathrm{SD}_j, a_j) : b_j \in A\}.$$

For MetaEvaluator, the same A is shared across all reference models:

$$\mathcal{E}_A^{(m)} = \{(\mathrm{SD}_j^{(m)}, a_j^{(m)}) : b_j \in A\}, \qquad \mathcal{S}_A = \{\mathcal{E}_A^{(m)} : m \in \mathcal{M}\}.$$

Let g_A and g_B denote evaluators with the same architecture trained with supervision from A and B, respectively. Our goal is to preserve estimation accuracy using only a small fraction of the full workload pool:

$$\mathrm{MAE}(g_A) \approx \mathrm{MAE}(g_B) \quad \text{with} \quad |A| \ll |B|.$$

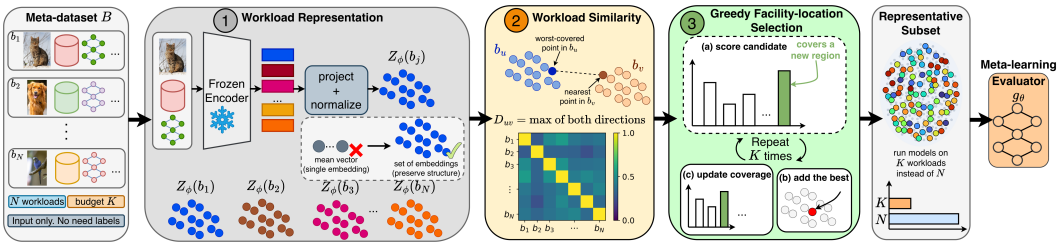

Figure 3: **Overview of ActiveEvaluator**. **(1)** Each workload in the meta-dataset is mapped to a finite embedding set. **(2)** Pairwise workload distances are converted into similarities. **(3)** Greedy facility location selects a representative subset $A \subseteq B$ under budget K. **(4)** The evaluator g_θ is trained on the selected meta-dataset and evaluated on unseen models.

The budget reduces the supervision cost from $|B|$ to $|A|$ model–workload runs for a model-specific evaluator, and from $R|B|$ to $R|A|$ for MetaEvaluator. The next section introduces the representativeness objective used to select A.

4 The ActiveEvaluator Method

Given a workload pool B and budget K, ActiveEvaluator selects a representative subset A before evaluator training, as illustrated in Fig. 3. Selection uses only workload embeddings and requires neither target labels nor reference-model runs. The selected subset can replace B when training either a model-specific evaluator, such as GNNEvaluator and FusionSQL, or MetaEvaluator, as formulated in §3.

4.1 Workload Representation

Standard facility location selects individual data points, but our ground-set elements are workloads containing many inputs. Representing each workload by a single pooled vector can hide important variation across its inputs. Therefore, we represent each workload as a set of embeddings and compare workloads at the set level.

For each input x, we serialize its content and encode it with a frozen contextual encoder. Let $h_r(x)$ denote the representation of the r-th retained token and W a fixed projection. We keep the normalized embedding of every token:

$$\phi(x) = \left\{ \frac{W h_r(x)}{\|W h_r(x)\|_2} : 1 \le r \le |x| \right\}, \tag{1}$$

where $|x|$ is the number of non-padding tokens after tokenization and standard encoder truncation. For a workload $b_j = \{x_1^j, \ldots, x_{n_j}^j\}$, we combine the embeddings of all its inputs:

$$Z_\phi(b_j) = \bigcup_{i=1}^{n_j} \phi(x_i^j). \tag{2}$$

Thus, $\phi(x)$ preserves the local structure of an individual input, while $Z_\phi(b_j)$ represents the workload as a whole. Importantly, this construction uses only the input data and requires neither labels nor model predictions.

4.2 Workload Similarity

Given two workloads b_u and b_v, we compare their embedding sets using the symmetric Hausdorff distance. We first define the directed distance:

$$d(b_u, b_v) = \max_{z \in Z_\phi(b_u)} \min_{z' \in Z_\phi(b_v)} \|z - z'\|_2,$$

which measures how far the least-covered embedding in b_u is from b_v. The symmetric Hausdorff distance is then:

$$D_{uv} = \max\{d(b_u, b_v), d(b_v, b_u)\}. \tag{3}$$

Algorithm 1: ActiveEvaluator representative selection

Require: meta-dataset $B = \{b_1, \ldots, b_N\}$, embedding map ϕ, budget K
// Stage 1: Workload representation
foreach $b_j \in B$ **do**
  compute $Z_\phi(b_j) = \bigcup_{i=1}^{n_j} \phi(x_i^j)$ (Eq. 2)
end
// Stage 2: Workload similarity
foreach $1 \le u < v \le N$ **do**
  compute $D_{uv} = D_{vu}$ from $Z_\phi(b_u)$ and $Z_\phi(b_v)$ (Eq. 3)
end
set $D_{uu} = 0$ for all u
set $\tau \leftarrow \text{median}_{u<v} D_{uv}^2$ and $S_{uv} \leftarrow \exp(-D_{uv}^2/\tau)$ for all u, v (Eq. 4)
// Stage 3: Greedy Facility-location Selection
$A \leftarrow \emptyset$
$c_j \leftarrow 0$ for all $j = 1, \ldots, N$
for $t = 1$ **to** $\min(K, N)$ **do**
  foreach $b_q \in B \setminus A$ **do**
    $\Delta_q \leftarrow \sum_{j=1}^{N} \max\{0, S_{qj} - c_j\}$ (Eq. 7)
  end
  $q^\star \leftarrow \arg\max_{q: b_q \notin A} \Delta_q$
  $A \leftarrow A \cup \{b_{q^\star}\}$
  for $j = 1$ **to** N **do**
    $c_j \leftarrow \max\{c_j, S_{q^\star j}\}$ (Eq. 5)
  end
end
return A

Unlike an average distance, D_{uv} is sensitive to embeddings that are poorly represented by the other workload. Therefore, it favors representatives that cover the workload as a whole rather than only its densest regions. This is later prove in §5.3.

We convert this distance into a non-negative similarity:

$$S_{uv} = \exp\left(-\frac{D_{uv}^2}{\tau}\right), \qquad \tau = \text{median}_{u<v} D_{uv}^2. \tag{4}$$

Thus, $S_{uv} \in (0, 1]$, with larger values indicating more similar workloads and $S_{uu} = 1$.

4.3 Greedy Facility-location Selection

Let $A \subseteq B$ denote the selected workloads. For each workload b_j, its coverage by A is the similarity to its best selected representative:

$$c_A(j) = \max_{b_u \in A} S_{uj}, \qquad c_\emptyset(j) = 0. \tag{5}$$

We score the selected set using the facility-location objective (Wei et al., 2015):

$$F(A) = \sum_{j=1}^{N} c_A(j) = \sum_{j=1}^{N} \max_{b_u \in A} S_{uj}.$$

ActiveEvaluator selects at most K workloads:

$$A^\star = \arg \max_{\substack{A \subseteq B \\ |A| \le K}} F(A). \tag{6}$$

This objective rewards coverage rather than redundancy. A candidate similar to workloads already represented by A provides little additional gain, whereas a candidate from a poorly covered region can substantially increase $F(A)$.

We optimize Eq. 6 greedily. Suppose A_t has been selected after t rounds. The marginal gain of adding an unselected workload b_q is:

$$\Delta(b_q \mid A_t) = \sum_{j=1}^{N} \max\left\{0,\ S_{qj} - c_{A_t}(j)\right\}. \tag{7}$$

Table 1: MAE in percentage points (mean ± 95% CI over 10 runs, lower is better). Pool size is 10 models. Each column is a target unlabeled dataset. Avg. is the mean across the seven datasets, with the CI averaged over columns. Best results are **bold** and second-best results are underlined.

Method	Spider	BIRD	Spider 2.0	BEAVER	ScienceBench	EntSQL	LiveSQLBench	Avg.	Meta-set Size
ConfProfile	8.38±1.01	9.66±1.48	12.02±1.65	11.13±1.58	10.44±1.50	11.04±1.64	11.36±1.62	10.58±1.50	-
ALine-D	8.41±1.02	9.74±1.31	11.68±1.70	10.89±1.75	10.11±1.43	10.44±1.65	11.32±1.87	10.37±1.53	-
NL2SQL-BUGs	8.15±1.19	9.37±1.44	11.42±1.60	11.09±1.53	10.05±1.55	10.66±1.40	11.49±1.59	10.32±1.47	-
ATC	7.86±1.04	9.48±1.36	11.81±1.65	10.94±1.61	9.70±1.52	10.49±1.42	11.31±1.60	10.23±1.46	-
DoC	7.78±1.00	9.38±1.28	11.48±1.69	10.56±1.67	9.68±1.55	9.92±1.54	10.89±1.47	9.96±1.46	-
GDE	7.66±1.03	8.95±1.35	11.20±1.89	10.54±1.71	9.72±1.54	9.84±1.49	11.12±1.80	9.86±1.54	-
FusionSQL	5.31±1.18	6.06±1.25	6.76±1.47	6.39±1.49	6.04±1.21	6.20±1.36	6.16±1.49	6.13±1.35	30K
MetaEvaluator	4.61±0.88	5.15±0.94	**6.32±0.99**	**6.08±0.96**	**5.62±0.91**	4.34±0.87	6.47±0.98	**5.51±0.93**	30K
ActiveEvaluator	**4.56±0.62**	**5.11±0.66**	6.55±0.69	6.32±0.68	5.85±0.67	**4.30±0.63**	**6.43±0.65**	5.59±0.66	9K

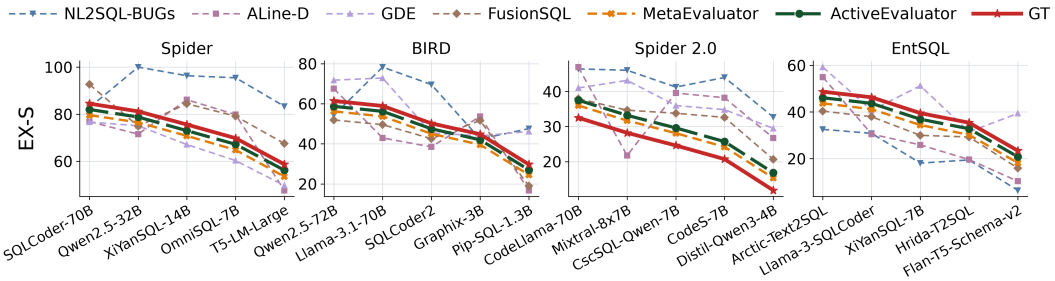

Figure 4: Calibration of accuracy estimation across four Text2SQL targets. Each panel plots average estimated EX-S against the ground truth (GT) for a pool of 10 models, showing 5 representative models per plot. Curves closer to GT indicate better-calibrated estimates.

At each round, we select the workload with the largest marginal gain and update the coverage of every workload. Alg. 1 summarizes the complete procedure. After K rounds, let A_K denote the subset returned by greedy selection. We show in Appendix A that F is normalized, monotone, and submodular. Therefore, greedy selection under the cardinality constraint satisfies:

$$F(A_K) \geq \left(1 - \frac{1}{e}\right) F(A^\star)$$

by the proof of Nemhauser et al. (1978). Thus, A_K attains at least $1 - 1/e$ of the optimal coverage achievable by any subset of at most K workloads from B.

5 Experiments

We focus on Text2SQL because its unbounded output space makes evaluation on unlabeled data particularly challenging. We use open-source models and datasets to ensure comparability with existing baselines. The evaluated model pool is listed in Appendix D. Results for image and node classification are reported in Appendix E and Appendix F, respectively.

5.1 Experimental Setup

Metrics and Implementation Details. We evaluate Text2SQL with EX-S (Appendix C), which requires the predicted and reference queries to agree on the original database and five variants. This reduces spurious matches accepted by single-instance execution accuracy (EX). Therefore, our EX-S MAE may differ from results reported solely under EX. Throughout the experiments, we report performance metrics, including MAE, EX-S, and end-to-end latency. We also report ranking-quality metrics, including pairwise ordering accuracy, Top-k selection accuracy, and Kendall's τ. All methods use the same source and target splits, model pool, evaluator architecture, and held-out target labels. Evaluator training and adaptation use bfloat16 mixed precision, parallel data loading, and fixed random seeds. Experiments run on four 24 GB NVIDIA RTX 4090 GPUs and an Intel Core i7-14700 CPU.

Datasets and Baselines. We evaluate on Spider (Yu et al., 2018), BIRD (Li et al., 2023), Spider 2.0 (Lei et al., 2025), BEAVER (Chen et al., 2024a), ScienceBenchmark (Zhang et al., 2023),

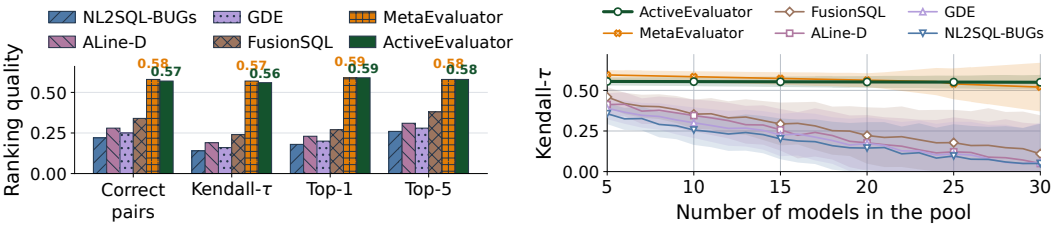

Figure 5: *Left*: ranking quality averaged over pools of 5 to 30 models. Higher is better for all four metrics. *Right*: Kendall's τ against pool size, with shaded 95% CIs over repeated runs.

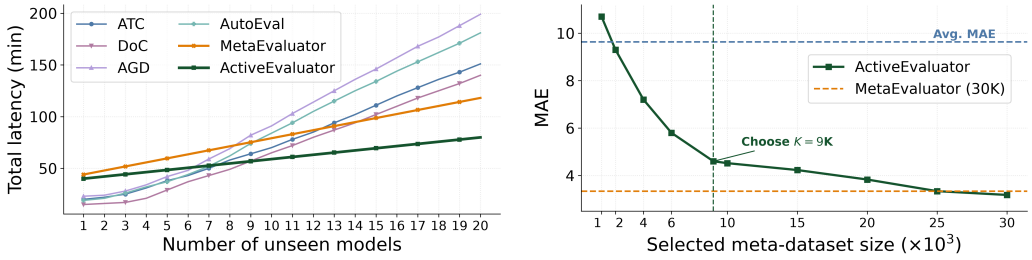

Figure 6: *Left*: total evaluation latency as the unseen-model pool grows. *Right*: MAE as the selected meta-dataset size increases. The blue line is the average MAE of baseline methods in Tab. 1. Lower is better in both panels.

EntSQL (Liao et al., 2026), and LiveSQLBench (BIRD Team, 2025). For each model, its training source is the closest compatible dataset among these benchmarks, rather than a workload from the meta-dataset. This assignment is shared by every method that quantifies the shift between model training data and an unlabeled target workload. Target labels remain hidden from every evaluator and are revealed only to compute the final metrics. We compare ActiveEvaluator with FusionSQL (Pham et al., 2026b), MetaEvaluator (Pham et al., 2026a), NL2SQL-BUGs (Liu et al., 2025b), GDE (Jiang et al., 2022), ALine-D (Baek et al., 2022), ATC (Garg et al., 2022), DoC (Guillory et al., 2021), and ConfProfile (Fu et al., 2023).

5.2 Performance Comparison

Evaluation Accuracy. As shown in Tab. 1, ActiveEvaluator outperforms MetaEvaluator on Spider, BIRD, EntSQL, and LiveSQLBench and achieves the narrowest 95% CIs across all datasets, while using only 9K of 30K workloads in the full meta-dataset. We attribute these gains on easier transfers to the effectiveness of representative selection, which removes redundant supervision and reduces variation across runs and model pools. In contrast, the full meta-dataset may retain duplicate or noisy workloads that bias the estimates downward. This pattern is visible in Fig. 4, where MetaEvaluator underestimates easier transfers more strongly than ActiveEvaluator. On harder benchmarks, including Spider 2.0, BEAVER, and ScienceBench, errors increase for both methods, yet ActiveEvaluator remains close to MetaEvaluator despite using only one third of the meta-dataset. Across models, scales, and datasets, ActiveEvaluator consistently preserves estimation quality, while other methods show larger and less systematic deviations that can distort pairwise differences and lead to ranking errors, as examined later. These experiments use a pool of 10 models sampled from Appendix D.

Ranking Stability. Fig. 5 shows that ActiveEvaluator preserves nearly the same ranking quality as MetaEvaluator with less than one third of the meta-dataset budget. MetaEvaluator performs slightly better in correct pairs and Kendall's τ, while both methods achieve the same Top-1 and Top-5 accuracy. As the pool grows from 5 to 30 models, ActiveEvaluator maintains stable Kendall's τ with narrow 95% CIs. MetaEvaluator starts higher but falls below ActiveEvaluator at 25 models, with noticeably wider CIs beyond 20 models, while the other baselines degrade more substantially. We attribute this behavior to redundant and noisy supervision in the full meta-dataset, whose influence can become stronger as more models are compared, disrupting knowledge transfer and relative model ordering. Stable ranking is important on unlabeled data, as this enables reliable model selection even when absolute estimates are imperfect.

Table 2: *Left*: impact of the meta-learning algorithm on MAE (95% CI) and adaptation steps. *Right*: impact of workload distance on MAE (95% CI) and runtime relative to Hausdorff.

Method	MAE	# Steps
MAML (Finn et al., 2017)	11.63 ± 1.21	12
FO-MAML (Finn et al., 2017)	8.88 ± 1.29	10
Reptile (Nichol et al., 2018)	9.12 ± 1.34	9
Meta-SGD (Li et al., 2017)	8.21 ± 1.18	9
ANIL (Raghu et al., 2020)	8.47 ± 1.25	8
ProtoNet (Snell et al., 2017)	12.45 ± 1.41	0
MetaEvaluator	$\mathbf{4.35 \pm 0.96}$	**5**
ActiveEvaluator	4.46 ± 0.78	**5**

Distance	MAE (95% CI)	Relative runtime
Hausdorff (Ours)	$\mathbf{3.47 \pm 0.74}$	$1.00\times$
Sliced Wasserstein	6.18 ± 2.12	$1.82\times$
Fréchet	6.74 ± 2.28	$1.31\times$
Chamfer	7.31 ± 2.37	$0.84\times$
Mahalanobis	8.14 ± 2.51	$1.55\times$
Euclidean	9.42 ± 2.86	$0.18\times$

5.3 Evaluation Efficiency

Evaluation Latency. Meta-learning first constructs supervision across a pool of models and workloads. Therefore, MetaEvaluator and ActiveEvaluator have higher initial latency than model-specific baselines in panel (a) of Fig. 6. However, this cost is paid once and amortized across future evaluations, for which both methods use five steps to adapt the context vector to each unseen model, as shown in Tab. 2. ActiveEvaluator attains comparable MAE with a narrower 95% confidence interval. Because it adapts over only the representative subset, its latency grows more slowly than MetaEvaluator as the model pool expands. This is shown later in the ablation study.

Meta-dataset Size. Neural-network error generally decreases as training data increase (Hestness et al., 2017; Kaplan et al., 2020). Consistent with this trend, panel (b) of Fig. 6 shows high MAE under very small budgets, while the blue reference line denotes the average MAE of all methods in Tab. 1. Beyond $K = 9$K, less than one third of the full 30K meta-dataset, additional workloads yield smaller but consistent gains, with ActiveEvaluator reaching the full MetaEvaluator reference at 25K. We attribute this efficiency to representative selection, which preserves broad workload coverage, and meta-learning, which transfers knowledge from limited supervision (Finn et al., 2017; Snell et al., 2017). Therefore, a well-covered subset can retain effective meta-knowledge without requiring the full meta-dataset. More generally, K can be chosen for a target MAE to balance evaluation cost and accuracy.

Ablation Study. The left panel of Tab. 2 shows that context-vector adaptation reaches lower MAE in fewer steps than weight-based adaptation. ActiveEvaluator closely matches MetaEvaluator with a narrower 95% CI while using only about one third of the full meta-dataset. As shown in Appendix B, this reduces computation under the same adaptation schedule while bounding the deviation from the full-data gradient. The right panel shows that Hausdorff distance achieves the lowest MAE and uncertainty while maintaining moderate runtime. Faster alternatives such as Euclidean and Chamfer reduce runtime but substantially increase MAE, whereas Sliced Wasserstein is both slower and less accurate. By preserving each workload as a set and emphasizing its least-covered region, Hausdorff provides the strongest balance between representative coverage, accuracy, and computational cost.

6 Conclusion

We study efficient evaluation of unseen models on unseen and unlabeled data, where transferable evaluators require costly supervision over large meta-datasets. ActiveEvaluator selects a small representative workload set under a fixed budget using Hausdorff similarity and facility-location coverage, with a $(1 - 1/e)$ greedy guarantee. Across Text2SQL, image classification, and node classification, it preserves MetaEvaluator's estimation and ranking quality with roughly one third of the meta-dataset, while reducing latency and uncertainty. As data and model pools continue to grow and inference becomes increasingly costly, ActiveEvaluator offers a practical direction for fast, budget-aware model evaluation when labeled data and computational resources are limited. More broadly, this suggests a shift from exhaustive benchmarking toward adaptive evaluation, where only the most informative workloads are acquired as new models and data arrive. Future work can extend this principle to dynamic budgets, continuously evolving model pools, and domains where supervision is difficult to obtain, making large-scale evaluation more practical as models and workloads continue to expand.

AI USE STATEMENT

We used generative AI to assist with language editing and presentation. The authors reviewed and verified all technical claims, proofs, citations, and experimental results and take full responsibility for the manuscript.

ETHICS STATEMENT

We evaluate Text2SQL, image classification, and node classification models using public benchmarks and open-source models. This study collects no new personal data and involves no human subjects. ActiveEvaluator reduces the resources required to evaluate models on unlabeled data, but its estimates may inherit biases from the underlying models, benchmarks, and workload embeddings. Therefore, estimated accuracy and rankings should support rather than replace labeled auditing in high-stakes deployments.

REPRODUCIBILITY STATEMENT

We define the evaluation setting and budgeted meta-dataset construction in §3. §4 specifies the workload representation, Hausdorff distance, facility-location objective, and greedy selection algorithm. Complete proofs are provided in Appendix A and Appendix B. The datasets, baselines, metrics, and implementation details are reported in §5.1, while the model pools and complementary results appear in Appendix D, Appendix E, and Appendix F. Our code is available at `https://anonymous.4open.science/r/ActiveEvaluator-00B1/`.

REFERENCES

Milad Aghajohari, Kamran Chitsaz, Amirhossein Kazemnejad, Sarath Chandar, Alessandro Sordoni, Aaron Courville, and Siva Reddy. The Markovian thinker: Architecture-agnostic linear scaling of reasoning. In *ICLR*, 2026.

Christina Baek, Yiding Jiang, Aditi Raghunathan, and J. Zico Kolter. Agreement-on-the-line: Predicting the performance of neural networks under distribution shift. In *NeurIPS*, 2022.

BIRD Team. LiveSQLBench: A dynamic and contamination-free benchmark for evaluating LLMs on real-world text-to-SQL tasks. `https://livesqlbench.ai/`, 2025.

Pierre Boyeau, Anastasios Nikolas Angelopoulos, Tianle Li, Nir Yosef, Jitendra Malik, and Michael I. Jordan. AutoEval done right: Using synthetic data for model evaluation. In *ICML*, 2025.

Mark Chen et al. Evaluating large language models trained on code. *arXiv preprint arXiv:2107.03374*, 2021.

Peter Baile Chen, Devin Yang, Weiyue Li, Fabian Wenz, Yi Zhang, Nesime Tatbul, Michael Cafarella, Çağatay Demiralp, and Michael Stonebraker. BEAVER: An enterprise benchmark for text-to-SQL. *arXiv preprint arXiv:2409.02038*, 2024a.

Runjin Chen, Tong Zhao, Ajay Kumar Jaiswal, Neil Shah, and Zhangyang Wang. LLaGA: Large language and graph assistant. In *ICML*, 2024b.

Xingyu Chen, Jiahao Xu, Tian Liang, Zhiwei He, Jianhui Pang, Dian Yu, Linfeng Song, Qiuzhi Liu, Mengfei Zhou, Zhuosheng Zhang, Rui Wang, Zhaopeng Tu, Haitao Mi, and Dong Yu. Do NOT think that much for 2+3=? on the overthinking of long reasoning models. In *ICML*, 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.

Weijian Deng and Liang Zheng. Are labels always necessary for classifier accuracy evaluation? In *CVPR*, 2021.

Weijian Deng, Yumin Suh, Stephen Gould, and Liang Zheng. Confidence and dispersity speak: Characterizing prediction matrix for unsupervised accuracy estimation. In *ICML*, 2023.

Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. In *ICLR*, 2021.

Ege Erdil. Inference economics of language models. *arXiv preprint arXiv:2506.04645*, 2025.

Chelsea Finn, Pieter Abbeel, and Sergey Levine. Model-agnostic meta-learning for fast adaptation of deep networks. In *ICML*, 2017.

Harvey Fu, Qinyuan Ye, Albert Xu, Xiang Ren, and Robin Jia. Estimating large language model capabilities without labeled test data. In *Findings of EMNLP*, 2023.

Saurabh Garg, Sivaraman Balakrishnan, Zachary C. Lipton, Behnam Neyshabur, and Hanie Sedghi. Leveraging unlabeled data to predict out-of-distribution performance. In *ICLR*, 2022.

Shurui Gui, Xiner Li, Limei Wang, and Shuiwang Ji. GOOD: A graph out-of-distribution benchmark. In *NeurIPS*, 2022.

Devin Guillory, Vaishaal Shankar, Sayna Ebrahimi, Trevor Darrell, and Ludwig Schmidt. Predicting with confidence on unseen distributions. In *ICCV*, 2021.

Hans Gundlach, Jayson Lynch, Matthias Mertens, and Neil Thompson. The price of progress: Price performance and the future of AI. *arXiv preprint arXiv:2511.23455*, 2025.

Daniel Haas, Jiannan Wang, Eugene Wu, and Michael J. Franklin. CLAMShell: Speeding up crowds for low-latency data labeling. *PVLDB*, 2015.

Dan Hendrycks and Thomas Dietterich. Benchmarking neural network robustness to common corruptions and perturbations. In *ICLR*, 2019.

Dan Hendrycks, Steven Basart, Norman Mu, Saurav Kadavath, Frank Wang, Evan Dorundo, Rahul Desai, Tyler Zhu, Samyak Parajuli, Mike Guo, Dawn Song, Jacob Steinhardt, and Justin Gilmer. The many faces of robustness: A critical analysis of out-of-distribution generalization. In *ICCV*, 2021a.

Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. Measuring mathematical problem solving with the MATH dataset. In *NeurIPS*, 2021b.

Joel Hestness, Sharan Narang, Newsha Ardalani, Gregory Diamos, Heewoo Jun, Hassan Kianinejad, Md. Mostofa Ali Patwary, Yang Yang, and Yanqi Zhou. Deep learning scaling is predictable, empirically. *arXiv preprint arXiv:1712.00409*, 2017.

Zijin Hong, Zheng Yuan, Qinggang Zhang, Hao Chen, Junnan Dong, Feiran Huang, and Xiao Huang. Next-generation database interfaces: A survey of LLM-based text-to-SQL. *TKDE*, 2025.

Weihua Hu, Matthias Fey, Marinka Zitnik, Yuxiao Dong, Hongyu Ren, Bowen Liu, Michele Catasta, and Jure Leskovec. Open graph benchmark: Datasets for machine learning on graphs. In *NeurIPS*, 2020.

Jonathan J. Hull. A database for handwritten text recognition research. *TPAMI*, 1994.

Yiding Jiang, Vaishnavh Nagarajan, Christina Baek, and J. Zico Kolter. Assessing generalization of SGD via disagreement. In *ICLR*, 2022.

Peter Kairouz, H. Brendan McMahan, Brendan Avent, Aurélien Bellet, Mehdi Bennis, Arjun Nitin Bhagoji, Kallista Bonawitz, Zachary Charles, Graham Cormode, Rachel Cummings, et al. Advances and open problems in federated learning. *Found. Trends Mach. Learn.*, 2021.

Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B. Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. Scaling laws for neural language models. *arXiv preprint arXiv:2001.08361*, 2020.

Sayash Kapoor, Benedikt Stroebl, Peter Kirgis, Nitya Nadgir, Zachary S. Siegel, Boyi Wei, Tianci Xue, Ziru Chen, Felix Chen, Saiteja Utpala, et al. Holistic agent leaderboard: The missing infrastructure for AI agent evaluation. In *ICLR*, 2026.

Manos Karpathiotakis, Vlassios Rizopoulos, Basri Kahveci, Tiziano Carotti, Artem Gelum, Hazem Nada, and Yuri Dolgov. Scribe: How Meta transports terabytes per second in real time. *PVLDB*, 2025.

Jiin Kim, Byeongjun Shin, Jinha Chung, and Minsoo Rhu. The cost of dynamic reasoning: Demystifying AI agents and test-time scaling from an AI infrastructure perspective. In *HPCA*, 2026.

Thomas N. Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. In *ICLR*, 2017.

Fangyu Lei, Jixuan Chen, Yuxiao Ye, Ruisheng Cao, Dongchan Shin, Hongjin Su, Zhaoqing Suo, Hongcheng Gao, Wenjing Hu, Pengcheng Yin, Victor Zhong, Caiming Xiong, Ruoxi Sun, Qian Liu, Sida Wang, and Tao Yu. Spider 2.0: Evaluating language models on real-world enterprise text-to-SQL workflows. In *ICLR*, 2025.

Haoyang Li, Jing Zhang, Hanbing Liu, Ju Fan, Xiaokang Zhang, Jun Zhu, Renjie Wei, Hongyan Pan, Cuiping Li, and Hong Chen. CodeS: Towards building open-source language models for text-to-SQL. *PACMMOD*, 2024.

Haoyang Li, Shang Wu, Xiaokang Zhang, Xinmei Huang, Jing Zhang, Fuxin Jiang, Shuai Wang, Tieying Zhang, Jianjun Chen, Rui Shi, Hong Chen, and Cuiping Li. OmniSQL: Synthesizing high-quality text-to-SQL data at scale. *PVLDB*, 2025.

Jinyang Li, Binyuan Hui, Ge Qu, Jiaxi Yang, Binhua Li, Bowen Li, Bailin Wang, Bowen Qin, Ruiying Geng, Nan Huo, Xuanhe Zhou, Chenhao Ma, Guoliang Li, Kevin C.-C. Chang, Fei Huang, Reynold Cheng, and Yongbin Li. Can LLM already serve as a database interface? a big bench for large-scale database grounded text-to-SQLs. In *NeurIPS*, 2023.

Zhenguo Li, Fengwei Zhou, Fei Chen, and Hang Li. Meta-SGD: Learning to learn quickly for few-shot learning. *arXiv preprint arXiv:1707.09835*, 2017.

Chengxi Liao, Tao Xu, Zulong Chen, Chuanfei Xu, Yiyan Wang, Xinyun Wang, Yanlong Zhang, Xiaojun Chen, Zhibo Yang, and Zeyi Wen. EntSQL: A benchmark for grounding text-to-SQL in long-context enterprise knowledge. *arXiv preprint arXiv:2606.03363*, 2026.

Xinyu Liu, Shuyu Shen, Boyan Li, Peixian Ma, Runzhi Jiang, Yuxin Zhang, Ju Fan, Guoliang Li, Nan Tang, and Yuyu Luo. A survey of text-to-SQL in the era of LLMs: Where are we, and where are we going? *TKDE*, 2025a.

Xinyu Liu, Shuyu Shen, Boyan Li, Nan Tang, and Yuyu Luo. NL2SQL-BUGs: A benchmark for detecting semantic errors in NL2SQL translation. In *KDD*, 2025b.

George L. Nemhauser, Laurence A. Wolsey, and Marshall L. Fisher. An analysis of approximations for maximizing submodular set functions—I. *Math. Program.*, 1978.

Yuval Netzer, Tao Wang, Adam Coates, Alessandro Bissacco, Bo Wu, and Andrew Y. Ng. Reading digits in natural images with unsupervised feature learning. In *NeurIPS Workshop on Deep Learning and Unsupervised Feature Learning*, 2011.

Alex Nichol, Joshua Achiam, and John Schulman. On first-order meta-learning algorithms. *arXiv preprint arXiv:1803.02999*, 2018.

Trinh Pham, Viet Huynh, Hongzhi Yin, Quoc Viet Hung Nguyen, and Thanh Tam Nguyen. Learning to evaluate: Cost-effective model evaluation on unlabeled data with meta-learning. In *KDD*, 2026a.

Trinh Pham, Thanh Tam Nguyen, Viet Huynh, Hongzhi Yin, and Quoc Viet Hung Nguyen. An efficient and effective evaluator for Text2SQL models on unseen and unlabeled data. In *ICDE*, 2026b.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning transferable visual models from natural language supervision. In *ICML*, 2021.

Anirudh Raghu, Maithra Raghu, Samy Bengio, and Oriol Vinyals. Rapid learning or feature reuse? towards understanding the effectiveness of MAML. In *ICLR*, 2020.

Ladislav Rampasek, Mikhail Galkin, Vijay Prakash Dwivedi, Anh Tuan Luu, Guy Wolf, and Dominique Beaini. Recipe for a general, powerful, scalable graph transformer. In *NeurIPS*, 2022.

Benjamin Recht, Rebecca Roelofs, Ludwig Schmidt, and Vaishaal Shankar. Do CIFAR-10 classifiers generalize to CIFAR-10? *arXiv preprint arXiv:1806.00451*, 2018.

Benjamin Recht, Rebecca Roelofs, Ludwig Schmidt, and Vaishaal Shankar. Do ImageNet classifiers generalize to ImageNet? In *ICML*, 2019.

Elan Rosenfeld and Saurabh Garg. (almost) provable error bounds under distribution shift via disagreement discrepancy. In *NeurIPS*, 2023.

Jake Snell, Kevin Swersky, and Richard Zemel. Prototypical networks for few-shot learning. In *NeurIPS*, 2017.

Yang Sui, Yu-Neng Chuang, Guanchu Wang, Jiamu Zhang, Tianyi Zhang, Jiayi Yuan, Hongyi Liu, Andrew Wen, Shaochen Zhong, Na Zou, Hanjie Chen, and Xia Hu. Stop overthinking: A survey on efficient reasoning for large language models. *TMLR*, 2025.

Jiabin Tang, Yuhao Yang, Wei Wei, Lei Shi, Lixin Su, Suqi Cheng, Dawei Yin, and Chao Huang. GraphGPT: Graph instruction tuning for large language models. In *SIGIR*, 2024.

Haohan Wang, Songwei Ge, Zachary C. Lipton, and Eric P. Xing. Learning robust global representations by penalizing local predictive power. In *NeurIPS*, 2019.

Kai Wei, Rishabh Iyer, and Jeff Bilmes. Submodularity in data subset selection and active learning. In *ICML*, 2015.

Jacob Whitehill, Paul Ruvolo, Tingfan Wu, Jacob Bergsma, and Javier Movellan. Whose vote should count more: Optimal integration of labels from labelers of unknown expertise. In *NeurIPS*, 2009.

Adina Yakefu, Apolinário, and Irene Solaiman. State of open models: Summer 2026 observations. Hugging Face, 2026.

Tao Yu, Rui Zhang, Kai Yang, Michihiro Yasunaga, Dongxu Wang, Zifan Li, James Ma, Irene Li, Qingning Yao, Shanelle Roman, Zilin Zhang, and Dragomir Radev. Spider: A large-scale human-labeled dataset for complex and cross-domain semantic parsing and text-to-SQL task. In *EMNLP*, 2018.

Xiaohua Zhai, Basil Mustafa, Alexander Kolesnikov, and Lucas Beyer. Sigmoid loss for language image pre-training. In *ICCV*, 2023.

Jintian Zhang, Yuqi Zhu, Mengshu Sun, Yujie Luo, Shuofei Qiao, Lun Du, Da Zheng, Huajun Chen, and Ningyu Zhang. LightThinker: Thinking step-by-step compression. In *EMNLP*, 2025.

Yi Zhang, Jan Deriu, George Katsogiannis-Meimarakis, Catherine Kosten, Georgia Koutrika, and Kurt Stockinger. ScienceBenchmark: A complex real-world benchmark for evaluating natural language to SQL systems. *PVLDB*, 2023.

Yuchen Zhang, Xi Chen, Dengyong Zhou, and Michael I. Jordan. Spectral methods meet EM: A provably optimal algorithm for crowdsourcing. *JMLR*, 2016.

Xin Zheng, Miao Zhang, Chunyang Chen, Soheila Molaei, Chuan Zhou, and Shirui Pan. GNNEvaluator: Evaluating GNN performance on unseen graphs without labels. In *NeurIPS*, 2023.

APPENDIX CONTENTS

A Monotonicity and Submodularity **15**

B Gradient Approximation and Computational Efficiency **16**

C EX-S **18**

D Text2SQL Model Pool **18**

E Image Classification **18**

F Node Classification **21**

A MONOTONICITY AND SUBMODULARITY

Let $B = \{b_1, \ldots, b_N\}$ be the full meta-dataset and let $A \subseteq B$ be the selected subset. The workload map Z_ϕ from Eq. 2 induces the non-negative similarity:

$$S_{uv} = \exp\left(-D_{uv}^2/\tau\right).$$

For $c_A(v) = \max_{b_u \in A} S_{uv}$ and $c_\emptyset(v) = 0$, the representative-selection objective is:

$$F(A) = \sum_{v=1}^{N} c_A(v) = \sum_{v=1}^{N} \max_{b_u \in A} S_{uv}. \tag{8}$$

Monotonicity. Let $A \subseteq C \subseteq B$. For any workload b_v, the candidate similarities available under A form a subset of those available under C:

$$\{S_{uv} : b_u \in A\} \subseteq \{S_{uv} : b_u \in C\}.$$

Taking a maximum over the larger set cannot decrease its value. Therefore, $c_A(v) \leq c_C(v)$ for every v. Summing these inequalities over the full meta-dataset gives:

$$F(A) = \sum_{v=1}^{N} c_A(v) \leq \sum_{v=1}^{N} c_C(v) = F(C).$$

Hence, adding workloads to the selected subset never decreases F, so F is monotone.

Submodularity. Let $A \subseteq C \subseteq B$ and let $b_q \in B \setminus C$. Adding b_q updates the coverage of each workload b_v as:

$$c_{A\cup\{b_q\}}(v) = \max\{c_A(v), S_{qv}\}.$$

The marginal gain of b_q relative to A is therefore:

$$\begin{aligned}\Delta(b_q \mid A) &= F(A \cup \{b_q\}) - F(A) \\ &= \sum_{v=1}^{N} [\max\{c_A(v), S_{qv}\} - c_A(v)] \\ &= \sum_{v=1}^{N} \max\{0, S_{qv} - c_A(v)\}.\end{aligned}$$

From monotonicity of the coverage maximum, $c_A(v) \leq c_C(v)$ for every v. For fixed S_{qv}, the function $r \mapsto \max\{0, S_{qv} - r\}$ is non-increasing. Thus:

$$\max\{0, S_{qv} - c_A(v)\} \geq \max\{0, S_{qv} - c_C(v)\}$$

for every v. Summing over all workloads yields:

$$\Delta(b_q \mid A) \geq \Delta(b_q \mid C).$$

Therefore, the gain from adding b_q cannot increase as the selected subset grows. This is the diminishing-returns condition, so F is submodular.

Assignment. After selecting A, we assign every workload b_v to its nearest selected representative:

$$u_A(v) \in \arg\min_{u:b_u \in A} D_{uv}. \tag{9}$$

Ties are resolved deterministically. Each selected workload b_u receives the multiplicity:

$$\gamma_u = \left|\{v \in \{1, \ldots, N\} : u_A(v) = u\}\right|. \tag{10}$$

The evaluator loss of b_u is weighted by γ_u, preserving the contribution of the workloads assigned to it. The weights satisfy $\sum_{u:b_u \in A} \gamma_u = N$. In Appendix B, this weighted objective connects geometric coverage to gradient approximation and computational efficiency.

Coverage. Define the distance from workload b_v to its nearest selected representative and the covering radius of A as:

$$d_A(v) = D_{u_A(v),v} = \min_{b_u \in A} D_{uv}, \qquad \varepsilon(A) = \max_{1 \leq v \leq N} d_A(v).$$

Because $t \mapsto \exp(-t^2/\tau)$ is decreasing for $t \geq 0$, the representative with the largest similarity is exactly the representative with the smallest distance. Explicitly:

$$\begin{aligned}\max_{b_u \in A} S_{uv} &= \max_{b_u \in A} \exp\left(-\frac{D_{uv}^2}{\tau}\right) \\ &= \exp\left(-\frac{\left[\min_{b_u \in A} D_{uv}\right]^2}{\tau}\right) \\ &= \exp\left(-\frac{d_A(v)^2}{\tau}\right).\end{aligned}$$

Substituting this identity into Eq. 8 gives:

$$F(A) = \sum_{v=1}^{N} \exp\bigl(-d_A(v)^2/\tau\bigr)\,.$$

Since $d_A(v) \leq \varepsilon(A)$ for every v, this identity gives $F(A) \geq N \exp(-\varepsilon(A)^2/\tau)$. A converse bound follows from the coverage deficit $\delta(A) = N - F(A)$. Every term in:

$$\delta(A) = \sum_{v=1}^{N} \left[1 - \exp\bigl(-d_A(v)^2/\tau\bigr)\right]$$

is non-negative. Therefore, for an index $v^\star$ satisfying $d_A(v^\star) = \varepsilon(A)$, we have $1 - \exp(-\varepsilon(A)^2/\tau) \leq \delta(A)$. Whenever $\delta(A) < 1$, rearranging yields:

$$\varepsilon(A) \leq \sqrt{\tau \log \frac{1}{1 - \delta(A)}}.$$

Therefore, ActiveEvaluator maximizes $F(A)$ to minimize the coverage deficit and obtain a smaller certified upper bound on the distance from every workload in B to its nearest selected workload in A. This establishes that the reduced meta-dataset A remains geometrically representative of the full workload pool B before evaluator training.

Transfer. To connect geometric coverage with model performance, we make a smoothness assumption. For the fixed model f from §3, let $a_f(b_v) = a_v \in [0, 1]$ denote its true performance on workload b_v. We assume that nearby workloads have similar performance. Formally, a_f is L-Lipschitz under the workload distance:

$$|a_f(b_u) - a_f(b_v)| \leq L D_{uv} \qquad \text{for all } b_u, b_v \in B.$$

The constant L bounds the largest possible change in performance per unit of workload distance. Thus, a smaller L means that performance varies more smoothly across nearby workloads. For each workload b_v, let $u_A(v)$ be its nearest selected representative from Eq. 9. By the definition of $d_A(v)$, we have:

$$\left|a_f(b_v) - a_f(b_{u_A(v)})\right| \leq L d_A(v) \leq L\varepsilon(A). \tag{11}$$

Taking the maximum over v and substituting the coverage-radius bound gives:

$$\max_{1 \leq v \leq N} \left|a_f(b_v) - a_f(b_{u_A(v)})\right| \leq L\varepsilon(A) \leq L\sqrt{\tau \log \frac{1}{1 - \delta(A)}}.$$

Thus, maximizing the facility-location objective, or equivalently minimizing the coverage deficit $\delta(A)$, tightens a worst-case bound on the performance gap between every workload and its nearest selected representative. This explains why geometric coverage is a useful surrogate for representative selection when performance changes smoothly across nearby workloads.

B Gradient Approximation and Computational Efficiency

We now show how representative coverage preserves the training signal while reducing the cost of evaluator optimization.

Training objectives. Consider a differentiable evaluator-training or context-adaptation stage whose objective decomposes over model–workload pairs. Let θ collect the parameters to be optimized in this stage. For model $m \in \mathcal{M}$ and workload b_v, define its component loss as:

$$\ell_{m,v}(\theta) = \mathcal{L}\Big(g_\theta(\mathrm{SD}_v^{(m)}), a_v^{(m)}\Big).$$

The normalized objective over the full meta-dataset is:

$$J_B(\theta) = \frac{1}{RN} \sum_{m\in\mathcal{M}} \sum_{v=1}^{N} \ell_{m,v}(\theta). \tag{12}$$

Using the multiplicities from Eq. 10, the weighted objective over the selected subset is:

$$J_A(\theta) = \frac{1}{RN} \sum_{m\in\mathcal{M}} \sum_{u:b_u\in A} \gamma_u \ell_{m,u}(\theta). \tag{13}$$

Because $\sum_{u:b_u\in A} \gamma_u = N$, both objectives have the same normalization. The weight γ_u preserves the contribution of all workloads assigned to b_u without repeating their forward or backward passes.

Gradient regularity. Geometric similarity is useful for optimization only when nearby workloads induce similar training gradients. Recall that D_{uv} is the distance between workloads b_u and b_v from Eq. 3. Let $G \geq 0$ denote the maximum change in the evaluator gradient per unit of workload distance. We assume that, for every $m \in \mathcal{M}$, parameter vector θ, and pair $b_u, b_v \in B$:

$$\|\nabla\ell_{m,u}(\theta) - \nabla\ell_{m,v}(\theta)\|_2 \leq D_{uv}\, G. \tag{14}$$

Under the common assignment in Eq. 9, it immediately gives:

$$\big\|\nabla\ell_{m,v}(\theta) - \nabla\ell_{m,u_A(v)}(\theta)\big\|_2 \leq d_A(v)\, G \leq \varepsilon(A)\, G. \tag{15}$$

Thus, the same geometric coverage controls two different quantities. It bounds workload-performance transfer through L in Eq. 11 and gradient approximation through G in Eq. 15.

Gradient approximation. Each selected workload b_u represents γ_u workloads. Therefore, multiplying its gradient by γ_u is equivalent to contributing one copy of that gradient for every workload that is assigned to b_u. The gradient of Eq. 13 can thus be written as:

$$\nabla J_A(\theta) = \frac{1}{RN} \sum_{m\in\mathcal{M}} \sum_{v=1}^{N} \nabla\ell_{m,u_A(v)}(\theta). \tag{16}$$

Subtracting Eq. 16 from the gradient of Eq. 12 expresses the total discrepancy as the average of the gradient difference for each model–workload pair. These differences are vectors that may point in different directions. The triangle inequality provides an upper bound without requiring their directions to align by replacing the norm of their sum with the sum of their norms:

$$\begin{aligned}
\|\nabla J_B(\theta) - \nabla J_A(\theta)\|_2 &= \left\| \frac{1}{RN} \sum_{m\in\mathcal{M}} \sum_{v=1}^{N} \big[\nabla\ell_{m,v}(\theta) - \nabla\ell_{m,u_A(v)}(\theta)\big] \right\|_2 \\
&\leq \frac{1}{RN} \sum_{m\in\mathcal{M}} \sum_{v=1}^{N} \big\|\nabla\ell_{m,v}(\theta) - \nabla\ell_{m,u_A(v)}(\theta)\big\|_2 \\
&\leq \frac{1}{N} \sum_{v=1}^{N} d_A(v)\, G.
\end{aligned} \tag{17}$$

Here, D_{uv} is the distance between a particular pair of workloads, whereas $d_A(v)$ is the distance from b_v to its nearest selected representative. Since $u_A(v)$ identifies that representative, $D_{u_A(v),v} = d_A(v)$ for every v. Therefore, Eq. 14 bounds each gradient difference by $d_A(v)\, G$. This bound is independent of m, so summing over the R models produces a factor of R that cancels the same factor in the denominator. The remaining expression is G times the average nearest-representative distance over all workloads. Since every embedding in Eq. 1 is normalized, $0 \leq d_A(v) \leq 2$. The

function $x \mapsto 1 - \exp(-x)$ is concave on $[0, 4/\tau]$. Therefore, its value is at least the corresponding linear interpolation between the two endpoints of this interval. Substituting $x = d_A(v)^2/\tau$ gives:

$$1 - \exp\left(-\frac{d_A(v)^2}{\tau}\right) \geq \frac{1 - \exp(-4/\tau)}{4} d_A(v)^2. \tag{18}$$

Summing Eq. 18 over all workloads and applying the Cauchy–Schwarz inequality gives:

$$\begin{aligned} \frac{1}{N}\sum_{v=1}^{N} d_A(v) &\leq \sqrt{\frac{1}{N}\sum_{v=1}^{N} d_A(v)^2} \\ &\leq 2\sqrt{\frac{\delta(A)}{N\left(1 - \exp(-4/\tau)\right)}}. \end{aligned} \tag{19}$$

Combining Eq. 17 and Eq. 19 yields the gradient approximation guarantee:

$$\|\nabla J_B(\theta) - \nabla J_A(\theta)\|_2 \leq 2G\sqrt{\frac{\delta(A)}{N\left(1 - \exp(-4/\tau)\right)}}. \tag{20}$$

Therefore, maximizing facility-location coverage reduces a certified upper bound on the discrepancy between the selected and full meta-training gradients. This result shows that representative selection preserves the full-data training signal. The computation count below separately establishes how much training cost is reduced.

Computational gain. Evaluating the full gradient in Eq. 12 requires RN model–workload components, whereas Eq. 13 requires only $R|A| \leq RK$. Therefore, each optimization step uses at most a fraction K/N of the full computation. The absolute reduction is $R(N - |A|)$ components per step, which is at least $R(N-K)$. The relative reduction is independent of the number of models, while the saved computation grows linearly as the model pool expands. Each additional model avoids at least $N - K$ model–workload components when the same representative subset is reused. Therefore, ActiveEvaluator reduces the cost of each optimization step while Eq. 20 controls its discrepancy from the full-data gradient.

C EX-S

Execution accuracy on a single database can accept an incorrect SQL query when its output coincides with that of the reference query. EX-S instead compares both queries on the original database and five controlled variants whose construction is guided by GPT-5.4. In Tab. 3, both queries return Mia and Noah on D, so standard EX incorrectly accepts the prediction. The first variant adds a Data Science student below the predicted credit range, the second adds a Design student within that range, and the third moves Noah's credits beyond the upper bound. These changes expose three distinct disagreements, while the remaining variants remove rows without changing the result. Because EX-S requires agreement on every instance, it correctly rejects the prediction.

D TEXT2SQL MODEL POOL

We form a fixed pool of $H = 35$ models from Hugging Face, listed in Tab. 4. We exclude quantized mirrors and duplicate checkpoints. Parameter counts come from the corresponding model pages, with a dash denoting an unreported count.

E IMAGE CLASSIFICATION

Datasets and Baselines. We evaluate on USPS (Hull, 1994), SVHN (Netzer et al., 2011), CIFAR-10.1 (Recht et al., 2018), CIFAR-10-C (Hendrycks & Dietterich, 2019), ImageNet-V2 (Recht et al., 2019), ImageNet-R (Hendrycks et al., 2021a), and ImageNet-Sketch (Wang et al., 2019), using MNIST, CIFAR-10, and ImageNet as the matching sources. Each target uses a compatible class vocabulary, with labels hidden from all evaluators. We compare ActiveEvaluator with baselines,

Original database (D)

name	program	credits
Mia	Data Science	60
Noah	Data Science	78
Olivia	Design	36
Parker	Biology	108

Instance 1

name	program	credits
Mia	Data Science	60
Noah	Data Science	78
Olivia	Design	36
Parker	Biology	108
Quinn	Data Science	24

Instance 2

name	program	credits
Mia	Data Science	60
Noah	Data Science	78
Olivia	Design	36
Parker	Biology	108
Riley	Design	54

Instance 3

name	program	credits
Mia	Data Science	60
Noah	Data Science	96
Olivia	Design	36
Parker	Biology	108

Instance 4

name	program	credits
Noah	Data Science	78
Olivia	Design	36
Parker	Biology	108

Removed: Mia

Instance 5

name	program	credits
Olivia	Design	36
Parker	Biology	108

Removed: Mia, Noah

Question	Which students are in the Data Science program?
Reference	`SELECT name FROM student WHERE program = 'Data Science';`
Prediction	`SELECT name FROM student WHERE credits BETWEEN 45 AND 90;`

	Original D	Instance 1	Instance 2	Instance 3	Instance 4	Instance 5
Reference output	Mia, Noah	Mia, Noah, Quinn	Mia, Noah	Mia, Noah	Noah	∅
Prediction output	Mia, Noah	Mia, Noah	Mia, Noah, Riley	Mia	Noah	∅
Match	✓	✗	✗	✗	✓	✓

Table 3: A student database and five controlled variants for comparing standard EX with EX-S.

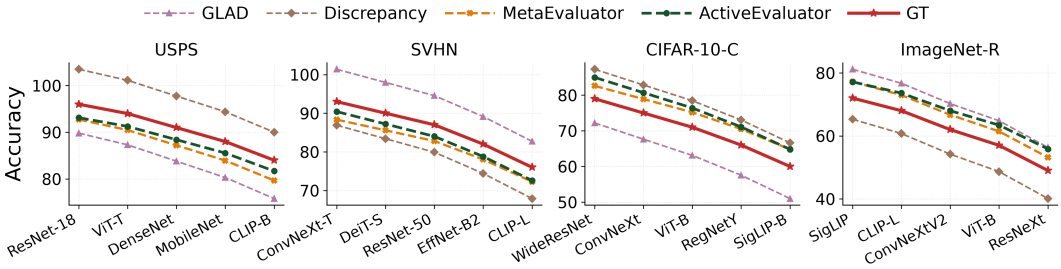

Figure 7: Estimated and ground-truth image classification accuracy across representative targets. Curves closer to ground truth indicate more accurate estimates.

including AutoEval (Deng & Zheng, 2021), MetaEvaluator (Pham et al., 2026a), GDE (Jiang et al., 2022), ALine-D (Baek et al., 2022), Discrepancy (Rosenfeld & Garg, 2023), ATC (Garg et al., 2022), DoC (Guillory et al., 2021), GLAD (Whitehill et al., 2009), and SpectralDS (Zhang et al., 2016). The aggregation methods treat classifiers as annotators and infer their reliability from agreement on shared predictions.

Model Pools. The pool holds 20 backbones, listed in Tab. 5: ten from the architecture families used by MetaEvaluator (Pham et al., 2026a), six widely downloaded Hugging Face classification checkpoints, and four vision–language models (Radford et al., 2021; Zhai et al., 2023) that classify zero-shot by scoring the target class names. Every model predicts into the target class vocabulary, with classification heads matched to each source and target rather than shared across the digit, CIFAR, and ImageNet tasks.

Metrics and Implementation Details. Ground-truth performance is top-1 classification accuracy, and MAE is reported in percentage points. Ranking quality follows the metrics in §5.1. All methods share the same source and target splits, model pool, evaluator architecture, and hidden target labels. We use the hardware, precision, and random-seed protocol of the main experiments.

Evaluation Accuracy. As shown in Tab. 6, ActiveEvaluator remains close to MetaEvaluator across digit, corruption, and rendition shifts while using 8K of the full 30K meta-dataset. It performs slightly better on the easier USPS, SVHN, CIFAR-10.1, and ImageNet-V2 targets, while MetaEvaluator retains a small advantage on the harder shifts. ActiveEvaluator also has the narrowest 95% CIs

Model	#Params	Link
Arctic-Text2SQL-R1-7B	7.6B	Snowflake/Arctic-Text2SQL-R1-7B
Code-Llama-2-13B-Instruct-Text2SQL	13.0B	bugdaryan/Code-Llama-2-13B-instruct-text2sql
CodeLlama-70B-Instruct	69.0B	codellama/CodeLlama-70b-Instruct-hf
CodeS-1B	1.0B	seeklhy/codes-1b
CodeS-3B	3.0B	seeklhy/codes-3b
CodeS-7B	7.0B	seeklhy/codes-7b
CodeS-15B	15.0B	seeklhy/codes-15b
CscSQL-Merge-Qwen2.5-Coder-7B	7.6B	cycloneboy/CscSQL-Merge-Qwen2.5-Coder-7B-Instruct
Distil-Qwen3-4B-Text2SQL	4.0B	distil-labs/distil-qwen3-4b-text2sql
Flan-T5-Text2SQL-Schema	247.6M	juierror/flan-t5-text2sql-with-schema
Flan-T5-Text2SQL-Schema-v2	247.5M	juierror/flan-t5-text2sql-with-schema-v2
Gemma-2-2B	2.6B	google/gemma-2-2b
Graphix-3B	3.0B	patrickNLP/Graphix-3B
Hrida-T2SQL-3B	3.8B	HridaAI/Hrida-T2SQL-3B-V0.2
Llama-3-SQLCoder-8B	8.0B	defog/llama-3-sqlcoder-8b
Llama-3.1-70B-Instruct	70.6B	meta-llama/Llama-3.1-70B-Instruct
Meta-Llama-3-70B	70.6B	meta-llama/Meta-Llama-3-70B
Ministral-3-14B	13.9B	mistralai/Ministral-3-14B-Instruct-2512
Mixtral-8x7B-Instruct	46.7B	mistralai/Mixtral-8x7B-Instruct-v0.1
OmniSQL-7B	7.6B	seeklhy/OmniSQL-7B
OmniSQL-14B	14.8B	seeklhy/OmniSQL-14B
Pip-SQL-1.3B	1.3B	PipableAI/pip-sql-1.3b
Qwen2.5-32B	32.8B	Qwen/Qwen2.5-32B
Qwen2.5-72B-Instruct	72.7B	Qwen/Qwen2.5-72B-Instruct
Qwen3-4B-Text-to-SQL	4.0B	Ellbendls/Qwen-3-4b-Text_to_SQL
SQLCoder-7B-2	6.7B	defog/sqlcoder-7b-2
SQLCoder-15B	15.5B†	defog/sqlcoder
SQLCoder-70B-Alpha	69.0B	defog/sqlcoder-70b-alpha
SQLCoder2	15.5B†	defog/sqlcoder2
T5-LM-Large-Text2SQL-Spider	783.2M	gaussalgo/T5-LM-Large-text2sql-spider
T5-Small-Text-to-SQL	60.5M†	cssupport/t5-small-awesome-text-to-sql
XiYanSQL-QwenCoder-3B	3.4B	XGenerationLab/XiYanSQL-QwenCoder-3B-2502
XiYanSQL-QwenCoder-7B	7.6B	XGenerationLab/XiYanSQL-QwenCoder-7B-2502
XiYanSQL-QwenCoder-14B	14.8B	XGenerationLab/XiYanSQL-QwenCoder-14B-2502
XiYanSQL-QwenCoder-32B	32.8B	XGenerationLab/XiYanSQL-QwenCoder-32B-2504

Table 4: Text2SQL model pool. Parameter counts are read from the Hugging Face model pages. Counts marked † are not reported on the model page and are derived from the sizes of the published weight files and their stored precision.

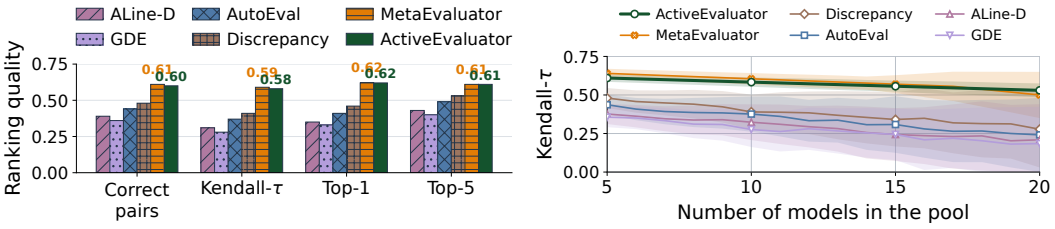

Figure 8: *Left*: ranking quality averaged over pools of 5 to 20 models. Higher is better for all four metrics. *Right*: Kendall's τ against pool size, with shaded 95% CIs over repeated runs.

across all targets. Fig. 7 shows the same behavior across architectures and model sizes, where both methods track ground-truth accuracy more consistently than the image-specific baselines.

Ranking Stability. Fig. 8 shows that ActiveEvaluator preserves nearly the same correct-pair, Kendall, Top-1, and Top-5 quality as MetaEvaluator using less than one third of its supervision. The gap remains small as the pool grows from 5 to 20 classifiers, while the other image-evaluation methods degrade more strongly. Therefore, representative selection retains the relative ordering needed to choose image models under a limited evaluation budget.

Model	#Params	Link
CLIP-ViT-B/32	151.3M†	openai/clip-vit-base-patch32
CLIP-ViT-L/14	427.6M	openai/clip-vit-large-patch14
ConvNeXt-Tiny	28.6M	timm/convnext_tiny.fb_in1k
ConvNeXtV2-Base	88.7M	facebook/convnextv2-base-22k-384
DeiT-Small	22.1M	timm/deit_small_patch16_224.fb_in1k
DenseNet-121	8.1M	timm/densenet121.ra_in1k
EfficientNet-B2	9.2M	timm/efficientnet_b2.ra_in1k
MobileNetV2	3.5M	timm/mobilenetv2_100.ra_in1k
MobileNetV3-Small	2.6M	timm/mobilenetv3_small_100.lamb_in1k
RegNetY-8GF	39.3M	timm/regnety_080.ra3_in1k
ResNet-18 – Microsoft	11.7M	microsoft/resnet-18
ResNet-18 – timm	11.7M	timm/resnet18.a1_in1k
ResNet-50 – Microsoft	25.6M	microsoft/resnet-50
ResNeXt-50-32x4d	25.1M	timm/resnext50_32x4d.a1_in1k
SigLIP2-Base/16	375.2M	google/siglip2-base-patch16-224
SigLIP-SO400M/14	878.0M	google/siglip-so400m-patch14-384
VGG-16	138.4M	timm/vgg16.tv_in1k
ViT-Base	86.6M	google/vit-base-patch16-224
ViT-Tiny	5.7M	timm/vit_tiny_patch16_224.augreg_in21k_ft_in1k
WideResNet-50-2	69.0M	timm/wide_resnet50_2.racm_in1k

Table 5: Image classification model pool. Parameter counts are read from the Hugging Face model pages, and counts marked † are derived instead from the published weight files.

Table 6: MAE in percentage points over unseen image-classification models (mean ± 95% CI over 10 runs, lower is better). Avg. is the mean across seven targets. Best results are **bold** and second-best results are underlined.

Method	USPS	SVHN	CIFAR-10.1	CIFAR-10-C	ImageNet-V2	ImageNet-R	ImageNet-Sketch	Avg.	Meta-set Size
ATC	9.05±1.48	12.41±1.85	10.48±1.76	12.66±2.12	11.54±1.78	13.91±2.48	13.27±2.36	11.90±1.98	–
DoC	8.42±1.24	11.96±1.74	9.13±1.40	11.88±1.62	10.37±1.39	12.78±1.82	13.29±2.03	11.12±1.61	–
GDE	7.74±1.66	10.39±2.31	9.48±2.02	10.72±2.14	9.91±2.16	12.54±2.71	12.83±2.18	10.52±2.17	–
AutoEval	7.65±1.22	10.28±1.51	8.27±1.08	10.35±1.54	9.16±1.25	11.24±1.61	11.66±1.79	9.80±1.43	–
ALine-D	6.51±1.31	9.31±2.08	7.75±1.59	9.22±1.83	8.39±1.67	10.57±2.26	12.08±2.19	9.12±1.85	–
SpectralDS	5.89±0.88	7.74±1.08	6.43±0.89	7.33±1.22	7.41±1.03	9.07±1.11	8.52±1.32	7.48±1.08	–
GLAD	5.32±1.18	7.12±1.54	5.96±1.22	7.61±1.47	7.05±1.31	7.96±1.78	8.91±1.88	7.13±1.48	–
Discrepancy	5.08±0.72	6.54±0.88	5.61±0.78	6.71±0.90	6.89±0.93	7.94±1.04	8.82±1.25	6.80±0.93	–
MetaEvaluator	2.83±0.92	3.49±0.98	3.09±0.91	**3.79±1.02**	3.39±0.94	**4.24±1.01**	**4.65±1.08**	**3.64±0.98**	30K
ActiveEvaluator	**2.77±0.62**	**3.43±0.66**	**3.03±0.61**	4.04±0.70	**3.33±0.64**	4.54±0.74	5.04±0.77	3.74±0.68	8K

Evaluation Latency. Image inference is less expensive than autoregressive Text2SQL generation, so the absolute latency in panel (a) of Fig. 9 is lower than in the main experiment. However, the scaling behavior remains consistent. MetaEvaluator and ActiveEvaluator pay a one-time meta-training cost, after which context adaptation is amortized across unseen classifiers. ActiveEvaluator uses only the selected workloads, so its total latency grows more slowly as the model pool expands.

Meta-dataset Size. Panel (b) of Fig. 9 shows that MAE decreases rapidly as representative image workloads are added. Beyond $K = 8$K, further supervision yields slower improvements because the selected set already covers the principal shifts. Therefore, 8K provides a favorable balance between accuracy and cost, while K can be increased when a lower target MAE is required.

F Node Classification

Datasets and Baselines. We evaluate on ACMv9, Citationv1, and DBLPv7 (Zheng et al., 2023), ogbn-arxiv (Hu et al., 2020), and GOOD-Cora, GOOD-Twitch, and GOOD-WebKB (Gui et al., 2022). These targets cover citation, social, and web-page graphs under distinct distribution shifts. Target-node labels remain hidden from every evaluator. We compare ActiveEvaluator with GNNEvaluator (Zheng et al., 2023), MetaEvaluator (Pham et al., 2026a), ALine-D (Baek et al., 2022), GDE (Jiang et al., 2022), Majority, and SpectralDS (Zhang et al., 2016).

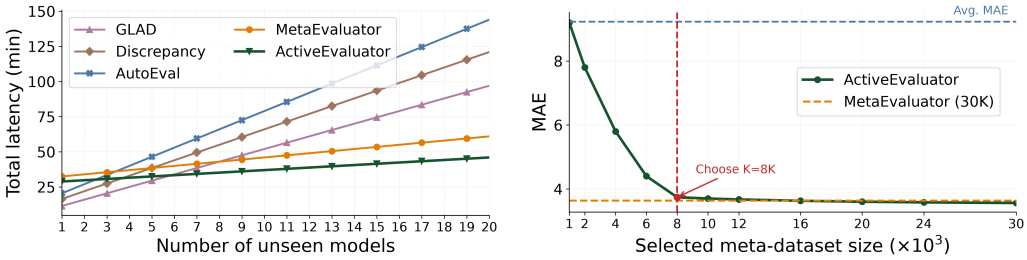

Figure 9: Image-classification efficiency under a reduced meta-dataset. *Left*: total latency as the unseen-model pool grows. *Right*: MAE as the selected meta-dataset expands, with $K = 8$K marking the operating budget. Lower is better in both panels.

Model	#Params	Link
APPNP	–	PyG APPNP
ARMA	–	PyG ARMAConv
ChebNet	–	PyG ChebConv
GAT	–	PyG GATConv
GATv2	–	PyG GATv2Conv
GCN	–	PyG GCNConv
GCNII	–	PyG GCN2Conv
GIN	–	PyG GINConv
GraphGPT-7B	6.9B†	Jiabin99/GraphGPT-7B-mix-all
GraphPFN-1.3	3.6M‡	eremeev-d/graphpfn-1.3
GraphSAGE	–	PyG SAGEConv
JKNet	–	PyG JumpingKnowledge
LLaGA-HO-Classification	6.7B‡	Runjin/llaga-vicuna-7b-simteg-HO-classification_expert-linear-projector
LLaGA-ND-Classification	6.7B‡	Runjin/llaga-vicuna-7b-simteg-ND-classification_expert-linear-projector
MixHop	–	PyG MixHopConv
MLP	–	PyG MLP
SGC	–	PyG SGConv
SuperGAT	–	PyG SuperGATConv
TAGCN	–	PyG TAGConv
UniMP	–	PyG TransformerConv

Table 7: Sixteen architectures are instantiated per target graph, so their parameter counts vary with the input dimension and the number of classes and are left unset. Counts marked † are derived from the published weight files and their stored precision. Models marked ‡ provide only adapter or projector weights. LLaGA is counted with its required Vicuna-7B-v1.5-16k base and GraphPFN by the adapter alone.

Model Pools. Each target draws on a compatible pool of 20 models listed in Tab. 7, collected through a broad search of relevant node-classification architectures and released graph checkpoints. The pool comprises 16 node-classification architectures, one graph foundation model, and three graph language models (Chen et al., 2024b; Tang et al., 2024) that answer node queries in text. The graph language models provide architectural diversity, and their generated answers are mapped to the target class vocabulary before agreement is computed. Architectures are trained per target with three seeds each, with labeled source nodes available wherever a baseline needs them for calibration.

Metrics and Implementation Details. Ground-truth performance is node-classification accuracy, and MAE is reported in percentage points. Ranking quality follows the metrics in §5.1. All methods share the same graph splits, model pool, evaluator architecture, and hidden target-node labels. We use the hardware, precision, and random-seed protocol of the main experiments.

Evaluation Accuracy. Tab. 8 shows that ActiveEvaluator closely approaches MetaEvaluator using 10K of the full 30K graph meta-dataset. It performs slightly better on ACMv9, Citationv1, and DBLPv7, while MetaEvaluator remains stronger on ogbn-arxiv and the harder GOOD shifts.

Table 8: MAE in percentage points over unseen node-classification models (mean ± 95% CI over 10 runs, lower is better). Avg. is the mean across seven targets. Best results are **bold** and second-best results are underlined.

Method	ACMv9	Citationv1	DBLPv7	ogbn-arxiv	GOOD-Cora	GOOD-Twitch	GOOD-WebKB	Avg.	Meta-set Size
Majority	10.31±1.44	10.84±1.49	11.68±1.39	13.77±1.84	13.64±1.65	16.12±1.91	16.05±2.28	13.20±1.71	–
SpectralDS	9.43±2.02	10.35±1.67	11.71±1.99	13.18±1.97	11.86±2.12	13.29±2.13	13.72±2.34	11.93±2.03	–
ALine-D	8.68±0.99	9.63±1.23	10.04±1.42	10.52±1.47	11.22±1.46	12.03±1.87	12.26±1.91	10.63±1.48	–
GDE	6.07±1.14	6.82±1.21	8.19±1.06	8.58±1.51	7.49±1.27	9.71±1.29	9.74±1.47	8.09±1.28	–
GNNEvaluator	4.78±1.31	5.31±1.24	6.21±1.68	6.36±1.66	5.78±1.39	7.31±2.03	7.62±1.94	6.20±1.61	–
MetaEvaluator	2.85±0.90	2.95±0.93	3.10±0.99	**3.40±1.06**	**3.55±0.98**	**3.85±1.08**	**4.10±1.13**	**3.40±1.01**	30K
ActiveEvaluator	**2.80±0.59**	**2.90±0.61**	**3.05±0.65**	3.75±0.70	3.93±0.68	4.25±0.76	4.52±0.79	3.60±0.68	10K

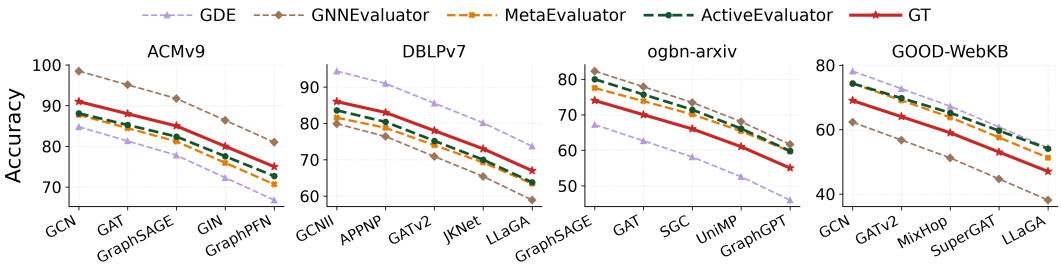

Figure 10: Estimated and ground-truth node classification accuracy across representative graph targets. Curves closer to ground truth indicate more accurate estimates.

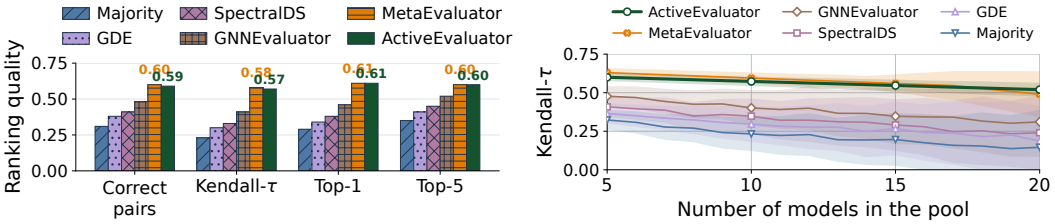

Figure 11: *Left*: ranking quality averaged over pools of 5 to 20 models. Higher is better for all four metrics. *Right*: Kendall's τ against pool size, with shaded 95% CIs over repeated runs.

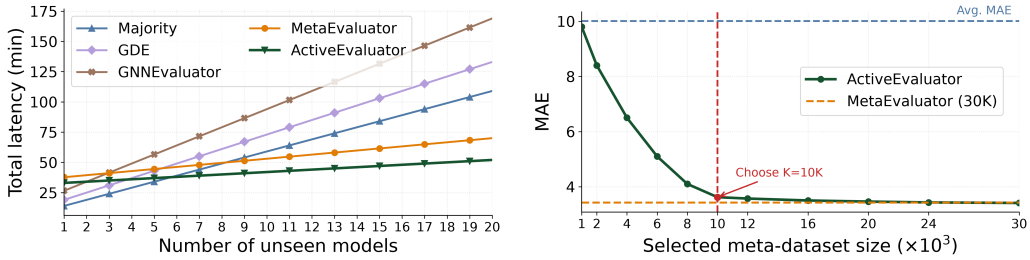

Figure 12: Node-classification efficiency under a reduced meta-dataset. *Left*: total latency as the unseen-model pool grows. *Right*: MAE as the selected meta-dataset expands, with $K = 10$K marking the operating budget. Lower is better in both panels.

ActiveEvaluator nevertheless achieves the narrowest 95% CIs throughout. As shown in Fig. 10, representative selection remains effective across message-passing networks, graph transformers, and graph-language models.

Ranking Stability. Fig. 11 shows that ActiveEvaluator preserves MetaEvaluator's ranking quality with one third of the graph meta-dataset. Its Kendall's τ remains stable as the pool grows from 5 to 20 models, while GNNEvaluator, agreement-based methods, and crowd aggregators decline more substantially. This indicates that representative graph workloads preserve the model ordering needed for budgeted deployment.

Evaluation Latency. Graph propagation and neighborhood aggregation produce a different absolute latency from image inference and Text2SQL generation. Panel (a) of Fig. 12 therefore uses the node-classification scale. ActiveEvaluator again grows more slowly than MetaEvaluator because each unseen model is adapted using only the representative graph workloads rather than the full meta-dataset.

Meta-dataset Size. Panel (b) of Fig. 12 shows rapid gains as the selected meta-dataset grows to $K = 10$K. Additional workloads yield smaller improvements beyond this point, indicating that the representative set already covers the major graph shifts. The 10K choice therefore retains nearly the full-data accuracy at one third of the supervision cost.