How Hard Can Indexing Be?
Principled Dataset Hardness Measurement for Learned Indexes

Siyuan Zhang
EPFL
siyuan.zhang@epfl.ch

Chuzhe Tang
EPFL
chuzhe.tang@epfl.ch

Anastasia Ailamaki
EPFL
anastasia.ailamaki@epfl.ch

ABSTRACT

Learned index structures have shown promising performance advantages over traditional indexes. Their performance is widely known to heavily depend on the underlying data distribution, as evidenced by anecdotes in the literature that certain datasets are "easy" or "hard" for learned indexes. However, a convincing quantitative metric of dataset hardness is still missing. As a result, practitioners cannot reliably estimate the potential benefits of learned indexes for their private workloads, and researchers have no principled way to identify blind spots in their index designs and evaluations.

This paper studies the problem of quantifying dataset hardness for learned indexes. First, we propose two scores, conformance and coverage, that together characterize metric quality while capturing a fundamental tradeoff between metric accuracy and applicability. Then, we evaluate several straightforward candidates and find that none is fully satisfactory. Still, our results reveal promising hardness indicators and offer guidelines for designing better metrics. Finally, we identify several open problems and future directions, including workload adaptation to circumvent this accuracy-applicability tradeoff in practice, and hardness-targeted dataset generation to fill the gaps in current learned index research and evaluation.

VLDB Workshop Reference Format:
Siyuan Zhang, Chuzhe Tang, and Anastasia Ailamaki. How Hard Can Indexing Be? Principled Dataset Hardness Measurement for Learned Indexes . VLDB 2026 Workshop: Applied AI for Database Systems and Applications (AIDB 2026).

1 INTRODUCTION

Efficient index structures are fundamental for high-performance data management systems. Learned index structures [9] have gained significant traction since their initial proposal in 2018, as they promise higher throughput, lower latency, and a smaller memory footprint than traditional indexes like B$^+$-trees. Core to the design of the initial proposal and all subsequent works [1–4, 6, 8, 11–13, 17, 24, 26, 29–33] is the idea of training machine learning models with the index working set to assist index lookup with prediction.

The performance of learned indexes is highly dependent on the underlying data distribution. For example, the throughput of ALEX, one of the representative learned indexes, can vary from 2.8 MOPS to 6.0 MOPS across different datasets [34, Fig. 8]. However, there are currently no principled guidelines for determining whether a dataset is learning-friendly or challenging. Existing studies mostly rely on a small set of commonly used datasets for evaluation. While this allows for easier comparison and cross-validation between different works, the lack of guidelines raises questions about the coverage these datasets provide across the spectrum of data distributions and how well learned indexes perform beyond them.

In this paper, we investigate: *How can we quantitatively measure dataset hardness, i.e., the difficulty of fitting a dataset with a learned index?* Dataset hardness has so far surfaced mostly anecdotally in the literature. Some works offer a qualitative ranking, e.g., "a synthetic dataset [`Linear`] and three real-world datasets [`Covid`, `Face`, `OSM`] with increasing fitting difficulty" [18]; others simply tag datasets with bare labels such as "`Libio` (easy)" or "`Genome` (hard)" without further elaboration [25]. To the best of our knowledge, only Wongkham et al. [27] have touched on hardness measurement, mentioning an intuitive design. However, there is no systematic validation of this proposal, so whether it is effective remains unclear. More fundamentally, how to properly judge the effectiveness of any hardness metric remains an open question.

First, we explore *what makes a hardness metric good?* Starting from a strawman proposal, we identify two quality scores, *conformance* and *coverage*, that together characterize the quality of a hardness metric. Conformance measures how well the hardness metric correlates with actual index performance across comparable datasets, while coverage is index-agnostic, measuring the proportion of datasets that are comparable under the hardness metric. Together, they capture the tradeoff between accuracy and applicability of a hardness metric. For better discriminative power, our scores place more emphasis on the challenging datasets that better differentiate candidate metrics, and less on the easy datasets that any reasonable metric should resolve correctly.

Then, we investigate *are straightforward metric designs good enough?* We evaluate 25 candidate metrics, including the one from Wongkham et al. [27], using 10 real-world datasets across 6 representative learned indexes. We find that none of these metric designs is fully satisfactory. First, no metric sits on or very close to the approximate conformance-coverage Pareto frontier, leaving a notable gap between the closest ones and the frontier. Second, every metric exhibits significant cross-index conformance variation; a metric with high overall conformance can still have very low conformance for specific indexes. Still, our results reveal several interesting findings, including promising hardness indicators and guidelines for designing better metrics, that can serve as a starting point for designing better metrics.

Finally, we consider *where do we go from here?* Building on these findings, we discuss several open problems and future directions. First, beyond what our greedy frontier approximation provides, we need a better understanding of how good a hardness metric

Proceedings of the VLDB Endowment. ISSN 2150-8097.

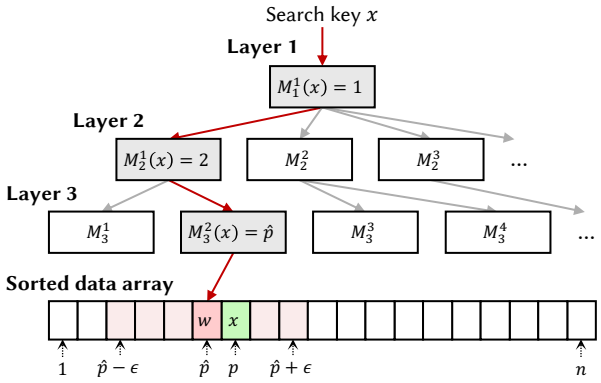

Figure 1: A 3-layer recursive model index.

can be, which would tell us how much headroom can be possibly closed. Second, beyond a set of near-Pareto-optimal metrics, we need a framework that adapts to user workloads, so that we can push the frontier outward beyond what any workload-agnostic metric can reach. Finally, we need dataset generation algorithms that reliably realize target hardness values across metrics, so we can fill the gaps in the hardness space and evaluate learned indexes more comprehensively.

In summary, our contributions are as follows:

- We identify the problem of evaluating the quality of dataset hardness metrics for learned indexes, and propose two intuitive quality scores, conformance and coverage.
- We find that no straightforward metric is fully satisfactory and present a number of findings, including effective hardness indicators and guidelines for designing better metrics.
- We discuss the implications of our findings and identify several open problems both in hardness metric design and, more broadly, in learned index design and evaluation.

2 BACKGROUND AND MOTIVATION

2.1 Learned Index Preliminaries

A learned index models the mapping from a key to its location in a sorted array, leveraging the fact that the empirical data distribution is often predictable. Let $S = \langle x_1 < \cdots < x_N \rangle$ be the sorted keys. The empirical cumulative distribution function (CDF) is $F(x) = |\{x_i \leq x\}|/N \in [0, 1]$. For indexing, we focus on the rank function $R(x) = |\{x_i \leq x\}| \in \{0, 1, \ldots, N\}$, which scales the CDF by N and directly gives the position of x in S. The initial learned index proposal introduces a multi-layer *recursive model index* (RMI) architecture, as shown in Figure 1, which is adopted and adapted by subsequent works. A root model M_1^1 at Level 1 routes a query key x to a specific model in the next layer. This process repeats through L layers; at each intermediate layer l, a model M_l evaluated at x selects a child in layer $l + 1$. Finally, a leaf model outputs a predicted position $\hat{p}$. At build time, models are trained on pairs $\{(x_i, R(x_i))\}_{i=1}^N$ to approximate $R(\cdot)$. At query time, the index traverses the hierarchy to obtain $\hat{p}$, and the exact position p is then recovered by a small verification search around $\hat{p}$, typically within a model-specific error window $[\hat{p} - \varepsilon, \hat{p} + \varepsilon]$ clipped to $[1, N]$. Usually, linear models are used for faster inference and smaller memory footprint.

Subsequent learned indexes preserve this core idea of approximating $R(\cdot)$ with a hierarchy of trained models, but differ in their specific index architectures. Unlike the original RMI, where each layer is a flat array of keys, later works explore more flexible tree organizations, where keys at the same layer may reside in different nodes. Different indexes build their trees with different partitioning strategies: error-driven greedy splits that bound the prediction error of each segment by a target ε (e.g., PGM-Index [2] and FITing-Tree [3]), even splits with error checking (e.g., XIndex [24]), cost-based splits that minimize an estimated lookup cost (e.g., ALEX [1]), and conflict-based splits that recurse on positions where multiple keys collide under the parent model (LIPP [29]). These strategies yield different tree shapes: balanced (e.g., PGM-Index and FITing-Tree), or unbalanced and adaptive, with deeper subtrees in regions that are harder to fit (e.g., ALEX and LIPP). They also differ in last-mile error handling. Most perform a local correction around $\hat{p}$ via either a bounded binary search or an exponential search, while LIPP eliminates correction entirely by storing each key in its predicted slot and chaining a new node whenever two keys collide. Because each of these choices interacts with the shape of $R(\cdot)$, the same dataset can be easy for one learned index and hard for another.

2.2 Measuring Dataset Hardness

While the performance of learned indexes is inherently dependent on the underlying datasets, there is currently no convincing quantitative measurement of dataset hardness, i.e., a hardness metric. A hardness metric $h(S)$ takes a dataset S and outputs a value, which potentially resides in a multidimensional space. If $h(S_1) < h(S_2)$, then S_1 is easier for learned indexes to fit and thus yields better performance than S_2. In this paper, we use the index throughput under a uniform read-only workload over the same dataset used to populate the index as the performance measurement.

Wongkham et al. [27] mentioned a straightforward dataset hardness metric, which we refer to as the `GRE` metric. They segment the dataset using piecewise linear approximation (PLA) [19] and propose to use the number of resulting segments as the hardness value. Specifically, two different error thresholds, 32 and 4096, are used to produce a local and a global dimension of hardness, denoted as h_l and h_g, respectively. PLA produces the theoretical minimum number of linear models required to fit a dataset under error constraints. Therefore, intuitively, the number of segments reflects overall learning difficulty.

However, they did not systematically validate the overall effectiveness of this metric, only reporting how h_l and h_g individually correlate with index performance on just two learned indexes. In fact, we find that it does not necessarily yield a good correlation in practice. For example, consider the `history` dataset ($h_l = 105k, h_g = 468$; see Table 1 for descriptions) and the `libio` dataset ($h_l = 146k, h_g = 639$). LIPP [29], a recent learned index, achieves 7.1 MOPS on the relatively hard `libio` dataset, but only 5.6 MOPS on the relatively easy `history` dataset.

This raises a question: is this an isolated violation, or does the `GRE` metric fail to correlate with learned index performance more generally? More fundamentally, for any given hardness metric, *how can*

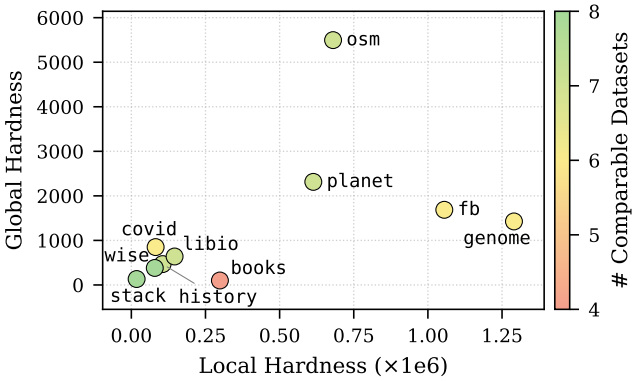

Figure 2: GRE hardness of different datasets, with data point color mapped to the number of comparable datasets each dataset has.

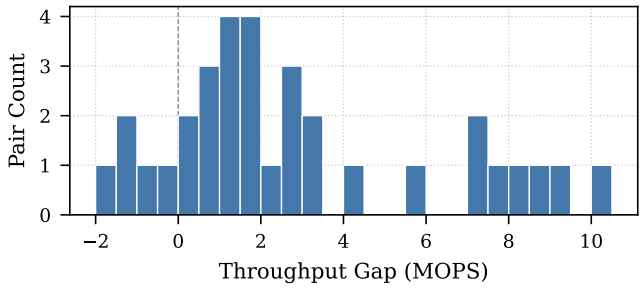

Figure 3: Distribution of the gap in LIPP's throughput across comparable dataset pairs. Negative values correspond to violating pairs.

we measure its effectiveness in capturing dataset hardness for learned indexes? In the rest of the paper, we propose two quality scores, conformance and coverage, and evaluate the GRE metric along with several other straightforward metrics under them. Based on our findings, we then discuss the implications and future directions.

3 METRIC CONFORMANCE AND COVERAGE

3.1 Strawman Measurement

Intuitively, to measure the quality of a hardness metric, given a dataset corpus and a learned index, we can look at the fraction of dataset pairs the metric orders correctly, i.e., a conformance score Conf = #conforming pairs/#total pairs. A conforming pair is a pair of datasets (S_1, S_2) such that if $h(S_1) < h(S_2)$, then the learned index performs better on S_1 than on S_2. With this definition, a score of one means total conformance and zero means total violation.

However, this conformance score is too coarse-grained to be informative. First, *it does not consider the comparability of dataset pairs.* A metric can be multidimensional, and thus does not necessarily totally order the dataset corpus. It is possible that, given two datasets S_1 and S_2, $h_i(S_1) < h_i(S_2)$ but $h_j(S_1) > h_j(S_2)$ for different dimensions i and j. Including incomparable pairs in the denominator mixes two aspects of metric quality, conformance and coverage, into one score that can be hard to interpret. As shown in Figure 2, with ten real-world datasets (Table 1), 27% (12/45) of the possible pairs are incomparable, spanning all ten datasets.

Second, *it does not differentiate the importance of individual comparable pairs.* Specifically, for conforming pairs, the ones with larger performance gaps are generally easier to order correctly. Therefore, they should contribute less to the score than pairs with smaller performance gaps. However, for violating pairs, those with larger performance gaps should weigh more heavily than those with smaller gaps, as they represent more egregious mistakes. As shown in Figure 3, comparable pairs differ greatly in their performance gaps.

3.2 Proposed Measurement

Given the observations above, we propose to measure the quality of a hardness metric with two separate scores, conformance and coverage. Both scores fall in $[-1, 1]$: for conformance, 1 means total conformance and -1 total violation; for coverage, 1 means every pair is comparable and -1 means none is. Keeping both scores bounded makes them easy to interpret and compare across metrics.

Notation. Consider a dataset corpus $\mathcal{S} = \{S_1, \ldots, S_n\}$, a learned index corpus $\mathcal{I} = \{I_1, \ldots, I_m\}$, and a d-dimensional hardness metric h. We use $p_I(S)$ to denote the performance of index $I \in \mathcal{I}$ on S, and σ_I^p to denote the standard deviation of index I's performance over $\mathcal{S}$. We use $h_k(S)$ to denote the k-th dimension of the hardness of dataset $S \in \mathcal{S}$. To put index performance on a comparable scale, we normalize it by its standard deviation, i.e., $\hat{p}_I(S) = p_I(S)/\sigma_I^p$, so that it has unit standard deviation. Given two datasets S_i and S_j, we say S_i is harder than S_j when $h_k(S_i) \geq h_k(S_j)$ for all k and $h_k(S_i) > h_k(S_j)$ for at least one k. Two datasets are *comparable* when one is harder than the other. We use $\mathcal{C} = \{(i, j) \mid S_i \text{ is harder than } S_j\}$ to denote all comparable dataset pairs as ordered pairs, and $\mathcal{U} = \{(i, j) \mid i < j \wedge (S_i \text{ and } S_j \text{ are incomparable})\}$ to denote the set of incomparable pairs. A comparable pair is *violating* for an index I when p_I is larger on the harder dataset, and *conforming* otherwise; this partitions $\mathcal{C}$ into a conforming set $\mathcal{C}_I^+$ and a violating set $\mathcal{C}_I^-$.

Conformance Score. We refine the strawman conformance by focusing only on comparable pairs and by further quantifying the degree of conformance or violation for each pair with its *importance.* The importance of a pair reflects how much we should care about ordering it correctly. From the signed performance gap $\delta p_{ij}^I = \hat{p}_I(S_i) - \hat{p}_I(S_j)$, we define the importance $w_{ij}^I = \text{sigmoid}(\delta p_{ij}^I)$, which maps the gap monotonically to a bounded importance in $(0, 1)$. For a conforming pair $(i, j) \in \mathcal{C}_I^+$, $\delta p_{ij}^I \leq 0$, and a larger gap means the index already separates the two datasets clearly, so the pair is easy to order correctly and should contribute little; for a violating pair $(i, j) \in \mathcal{C}_I^-$, $\delta p_{ij}^I > 0$, and a larger gap means a more egregious mistake, so the pair should weigh more heavily.

We accumulate the degree of conformance or violation of comparable pairs into a *reward* $R_I = \sum_{(i,j)\in\mathcal{C}_I^+} w_{ij}^I$ and a *penalty* $P_I = \sum_{(i,j)\in\mathcal{C}_I^-} w_{ij}^I$ for each index. The conformance score for an index is then their normalized difference, and the overall score averages it over all indexes,

$$\text{Conf}_I = \frac{R_I - P_I}{R_I + P_I}, \tag{1}$$

$$\text{Conf} = \text{mean}_{I\in\mathcal{I}}(\text{Conf}_I). \tag{2}$$

The normalized difference keeps the score within $[-1, 1]$ and comparable across indexes. A conformance of one means the metric perfectly conforms to the performance ordering of every index, while a conformance of minus one means every comparable pair violates it.

Coverage Score. Conformance only judges a metric on the pairs it can order, so a metric could score well by declaring most pairs incomparable and committing only to the safest few. Therefore, we define coverage to guard against this as follows.

$$\text{Cov} = \frac{|C| - |\mathcal{U}|}{|C| + |\mathcal{U}|} = \frac{|C| - |\mathcal{U}|}{N}, \tag{3}$$

where $N = \binom{n}{2}$ is the total number of dataset pairs. We introduce the $-|\mathcal{U}|$ term for consistency with the range of the conformance score, $[-1, 1]$, and thus ease of interpretation. A coverage of one means every pair is comparable under the metric, while a coverage of minus one means no pair is. Note that, unlike the conformance score, this coverage is independent of any index and thus reflects an intrinsic property of the metric itself.

4 EXPERIMENTAL RESULTS

In this section, we evaluate several hardness metrics through the lens of the conformance and coverage scores and analyze their fitness for learned indexes.

4.1 Setup

Metrics under Test. We evaluate the following hardness metrics.

- `RMSE` is the root mean square error of a linear model fitted on the dataset using least squares regression.
- `ME` is the maximum error of a linear model fitted on the dataset using least squares regression.
- `CD`, short for conflict degree, is the maximum number of keys mapped to the same rank by a linear model fitted on the dataset using Fastest Minimum Conflict Degree (FMCD) by Wu et al. [29]. It was originally proposed to quantify the model quality of LIPP, and thus has good correlation with LIPP's performance [10].
- `PLA-`ϵ is a scalar metric. It is the number of segments produced by PLA given a fixed error bound ϵ. It is used as the two components of the `GRE` metric, with ϵ values 32 and 4096. Therefore, as scalar metrics, we separately evaluate `PLA-32` and `PLA-4096`.
- `GRE` [27] is a two-dimensional metric. It uses `PLA-32` and `PLA-4096` as its local and global hardness dimensions, respectively.
- `A·B` is a two-dimensional metric. It composes two scalar metrics, A and B, chosen from the list above, to form the hardness score.
- `A·B·C` composes three scalar metrics, A, B, and C for a score.

All these scalar metrics have a time complexity of $O(N)$ to compute on sorted keys, where N is the number of keys in the dataset. In our evaluation, they typically take 10-20 seconds to compute for each dataset.

Learned Indexes. The following learned indexes are evaluated:

- `RMI` [9], as described in Section 2.1, is the initial learned index proposal that uses a recursive model architecture.
- `PGM-index` [2] uses a recursively PLA-constructed, optimal ϵ-bounded piecewise linear model to learn the rank function.
- `ALEX` [1] extends RMI with model-based inserts into gapped arrays and supports adaptive node expansion and splitting.

Table 1: Datasets used in our evaluation.

Dataset	Description
`books` [7]	Amazon book sales popularity.
`fb` [7]	Upsampled Facebook user ID.
`osm` [7]	Uniformly sampled OpenStreetMap locations.
`covid` [16]	Uniformly sampled Tweet ID with tag COVID-19.
`genome` [21]	Loci pairs in human chromosomes.
`history` [5]	History node ID in OpenStreetMap.
`libio` [14]	Repository ID from `libraries.io`.
`planet` [5]	Planet ID in OpenStreetMap.
`stack` [22]	Vote ID from StackOverflow.
`wise` [28]	Partition key from the data returned by the Wide-field Infrared Survey Explorer (WISE).

- `LIPP` [29] is a model-guided tree index that ensures precise key-to-position mappings without in-node search and achieves $O(\log N)$ lookup with amortized $O(\log^2 N)$ insert complexity.
- `XIndex` [24] uses range-partitioned groups indexed by linear models and a delta index, combined with fine-grained synchronization and compaction, to support dynamic workloads.
- `FINEdex` [11] organizes independently trained linear models as per-key two-level bins and supports in-place inserts and fine-grained concurrent retraining without a shared delta buffer.

Datasets. Table 1 summarizes the datasets used in our evaluation. To the best of our knowledge, they are a superset of the datasets used in recent work on learned indexes [4, 7, 8, 10, 12, 13, 17, 18, 23, 24, 27, 30, 32–34]. Each dataset contains 200 million 64-bit unsigned integer keys without duplicates.

Workloads and Measurements. We use index lookup throughput as the primary performance metric, as write operations introduce performance overhead that is not directly related to dataset hardness. For each index and dataset, we first bulk load the index and then perform random lookups for all keys in the dataset. For each run, we start with a warm-up phase of 20 million lookups and then a measurement phase of 100 million lookups, from which we calculate the throughput.

Testbed and Configuration. We use a server with two Intel Xeon Gold 5118 CPUs (12 cores per socket, 2.3 GHz) and 384 GiB of RAM for our evaluation. The container runs Ubuntu 20.04.6 with the host Linux kernel 6.8.0-57-generic. Hugepages are disabled. All indexes are ported in a common codebase and compiled with GCC 9.4.0 using the `-O3` flag. We use the default configuration for each index when applicable, and bug fixes are applied when necessary. The worker thread is pinned to a fixed CPU core for stability.

4.2 Results and Analysis

Conformance-Coverage Tradeoff. Figure 4 shows the relationship between the conformance and coverage scores of all the evaluated metrics. As can be seen, there is a clear tradeoff between conformance and coverage. As one side improves, the other side degrades, and there is no metric that simultaneously achieves ideal conformance and ideal coverage. This tradeoff is not surprising.

Table 2: The conformance and coverage of hardness metrics under test.

Metric	Conf_I						Conf	Cov
	RMI	PGM	ALEX	LIPP	XIndex	FINEdex		
Scalar metrics								
RMSE	0.34	0.19	0.41	0.09	−0.14	0.26	0.19	1.00
ME	0.32	0.04	0.39	0.22	−0.15	0.11	0.15	1.00
CD	0.18	0.36	0.10	0.84	0.09	0.11	0.28	1.00
PLA-32	0.41	0.64	0.71	0.29	0.02	0.91	0.50	1.00
PLA-4096	0.38	−0.03	0.17	0.03	0.03	−0.10	0.08	1.00
Two-dimensional metrics								
GRE (PLA-32·PLA-4096)	1.00	0.71	1.00	0.42	0.27	0.86	0.71	0.47
RMSE·ME	0.44	0.18	0.53	0.20	−0.09	0.24	0.25	0.82
RMSE·CD	0.68	0.69	0.69	0.89	0.14	0.53	0.60	0.47
RMSE·PLA-32	0.69	0.73	1.00	0.34	0.05	1.00	0.64	0.60
RMSE·PLA-4096	1.00	0.40	0.84	0.32	0.16	0.39	0.52	0.42
ME·CD	0.57	0.45	0.57	0.90	0.09	0.34	0.49	0.56
ME·PLA-32	0.70	0.62	1.00	0.43	0.04	0.88	0.61	0.60
ME·PLA-4096	1.00	0.30	0.85	0.42	0.14	0.29	0.50	0.42
CD·PLA-32	0.58	0.87	0.73	0.89	0.21	0.87	0.69	0.60
CD·PLA-4096	0.83	0.55	0.53	0.88	0.28	0.25	0.55	0.42
Three-dimensional metrics								
RMSE·ME·CD	0.67	0.68	0.68	0.88	0.13	0.51	0.59	0.42
RMSE·ME·PLA-32	0.67	0.71	1.00	0.39	0.01	1.00	0.63	0.51
RMSE·ME·PLA-4096	1.00	0.35	0.83	0.37	0.11	0.33	0.50	0.33
RMSE·CD·PLA-32	0.79	1.00	1.00	1.00	0.17	1.00	0.83	0.33
RMSE·CD·PLA-4096	1.00	0.76	0.77	1.00	0.14	0.53	0.70	0.16
RMSE·PLA-32·PLA-4096	1.00	0.80	1.00	0.42	0.15	1.00	0.73	0.24
ME·CD·PLA-32	0.81	0.83	1.00	1.00	0.18	0.84	0.78	0.38
ME·CD·PLA-4096	1.00	0.60	0.80	1.00	0.16	0.41	0.66	0.20
ME·PLA-32·PLA-4096	1.00	0.63	1.00	0.55	0.13	0.82	0.69	0.24
CD·PLA-32·PLA-4096	1.00	0.80	1.00	0.85	0.20	0.81	0.78	0.24

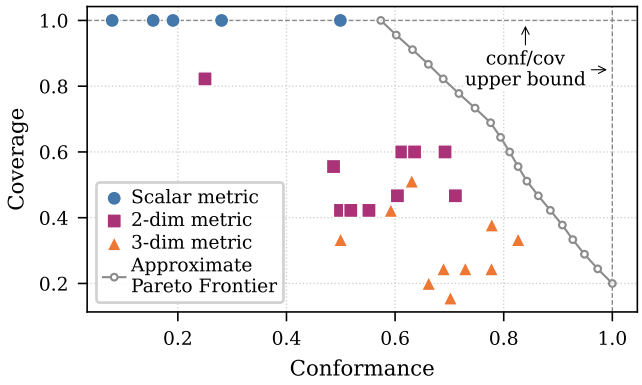

Figure 4: Overall conformance and coverage of hardness metrics under test. The ideal metric sits at the top-right corner.

A high-conformance metric can be achieved by tailoring a multi-dimensional metric where each dimension targets a specific learned index design of interest. However, with increased dimensionality, it becomes less likely two datasets have a harder-than relationship, leading to lower coverage. Meanwhile, a scalar metric achieves full coverage as it always totally orders all datasets.

> Finding 1. *There is a clear tradeoff between conformance and coverage in hardness metric design. Metric conformance generally increases with dimensionality, while coverage decreases.*

To understand this tradeoff better, we need the Pareto frontier in the conformance-coverage plane. We use a greedy algorithm inspired by beam search to approximate the exact frontier, which is computationally infeasible to calculate due to the combinatorial nature of the problem. We start with scalar metrics and use brute-force search over all possible orderings of datasets, encoded as integers ranging from 1 to 10 (the number of datasets) for the scalar hardness. This yields 10! = 3,628,800 candidates. Then, we select the top-1k metrics with the highest conformance scores and use them as the base metrics where another dimension is added to form two-dimensional metrics. Values for the added dimension are obtained by the same brute-force search over 3,628,800 candidates. Then, we select all the two-dimensional metrics on their Pareto frontier and use them as the base metrics to form three-dimensional metrics in a similar manner. This recursive process stops after exploring six-dimensional metrics, as it is guaranteed to achieve a conformance

of one with six dimensions, each corresponding to the ordering given by the performance of one index over a dataset. The final Pareto frontier, shown in Figure 4, is calculated from all the explored candidates. Optimizations that do not affect the final results are skipped for brevity.

Comparing the metrics under test with the approximate Pareto frontier, we find that none of the evaluated metrics sits on or very close to the frontier. Among all the metrics, PLA-32 is the closest scalar metric, CD·PLA-32 and GRE (i.e., PLA-32·PLA-4096) are the closest two-dimensional metrics, and RMSE·CD·PLA-32 and ME·CD·PLA-32 are the closest three-dimensional ones. PLA-32 appears in all of them as a hardness dimension, and CD appears in three. This suggests that they are more discriminative indicators of dataset hardness and combining them yields better conformance than others, which is further confirmed by our later analysis.

> Finding 2. *None of the evaluated metrics sits close to the approximate conformance-coverage Pareto frontier. Among the closest metrics,* PLA-32 *and* CD *are the common hardness dimensions and thus can serve as valuable starting points for designing better metrics.*

Scalar Metric Analysis. Next, we analyze individual metrics in more detail. Table 2 shows the conformance and coverage scores, as well as the conformance scores for individual indexes. We first focus on scalar metrics. CD achieves good conformance on LIPP, which is expected. LIPP constructs its tree structure based on the prediction conflicts and only incurs a performance penalty when accessing conflicting keys. CD directly reflects the amount of conflicts seen by the model at the root node, where the majority of conflicts happen, and therefore serves as a good performance predictor for LIPP.

Meanwhile, PLA-32 achieves good conformance on PGM-index, FINEdex, and ALEX. Unlike LIPP, these three indexes use cost-based partitioning of the key range and depend on local correction to handle model mispredictions. As a result, their performance is more influenced by the local linearity of the dataset, which is well captured by PLA-32. The more segments produced by PLA-32, the more locally non-linear the dataset is, leading to worse performance.

> Finding 3. *Different learned indexes require different hardness indicators that reflect their design principles.* CD *is a good fit for indexes with conflict-based designs, while* PLA-32 *suits indexes relying on local error correction.*

RMSE, ME, and PLA-4096 do not achieve good conformance with any of the six indexes. These metrics primarily capture the non-linearity of the dataset as a whole, which corresponds to neither the conflict-based design of LIPP nor the local correction-based design of PGM-index, FINEdex, and ALEX. This also explains why none of the three metrics achieves good conformance on RMI and XIndex. Instead of cost-based partitioning, RMI uses a fixed model architecture and relies on a single root linear model to dispatch keys to lower-level models, without considering the cost of mispredictions. Meanwhile, as one of the earliest learned indexes, XIndex simply reuses RMI to index its range-partitioned groups. As a result, its performance does not correlate well with any of these metrics.

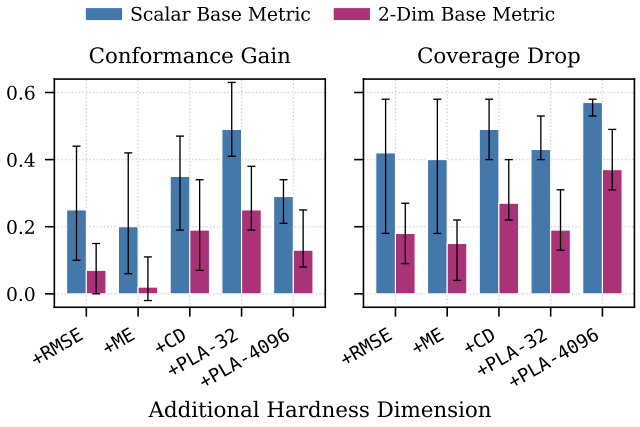

Figure 5: Impact of additional hardness dimensions on conformance and coverage scores. The gain (resp. drop) corresponds to the change in conformance scores (resp. coverage scores) when a new scalar metric (x axis) is added as a new dimension to another scalar or 2-dimensional base metric that does not already include it. Each bar represents the average score change; error bars span the minimum and maximum change.

> Finding 4. *Metrics that do not align with the design principles of any learned index do not achieve good conformance. None of the straightforward scalar metrics fits well with RMI and XIndex.*

Multi-Dimensional Metric Analysis. Finally, we move to multi-dimensional metrics. As mentioned earlier, increased dimensionality can improve conformance at the cost of coverage. Figure 5 quantifies this effect by showing the change in conformance and coverage when a new dimension is added on top of scalar base metrics (forming two-dimensional metrics) and of two-dimensional base metrics (forming three-dimensional metrics).

We observe a diminishing return effect on both conformance improvement and coverage degradation as dimensionality increases. In all cases, the change in both scores is more significant when going from one dimension to two dimensions than from two dimensions to three dimensions. We also note that adding new dimensions does not always improve conformance. For example, there are cases where adding RMSE or ME as the third dimension actually reduces conformance. These results suggest that two-dimensional metrics generally strike a good balance between conformance and coverage.

> Finding 5. *Introducing a new dimension to a hardness metric has a diminishing impact on both conformance improvement and coverage degradation as dimensionality increases. In some cases, going from two dimensions to three dimensions even reduces conformance.*

CD and PLA-32 are the most valuable dimensions to add in terms of conformance improvement. They achieve at least 2.4× and up to 11.0× conformance improvement compared to other scalar metrics as the added dimension. Meanwhile, other metrics, such as ME, can bring an average increase as low as 0.02 in conformance. For XIndex, since none of the scalar metrics fits well with it, as shown in Table 2,

combining them does not lead to significant conformance improvement, even for the most promising combination of CD and PLA-32. These results confirm our earlier finding that metrics should be carefully selected to align with the design principles of learned indexes to be effective.

> Finding 6. *Scalar metrics that align with the design principles of learned indexes are the most effective dimensions to add for conformance improvement. Combining misaligned scalar metrics, however, does not lead to significant conformance improvement.*

On the coverage side, no dimension degrades coverage notably more than the others, as the drop is similar in magnitude across all of them. Unlike conformance improvement that can be negative, coverage degradation is always positive, as expected, since adding a new dimension can only make some previously comparable pairs incomparable, but not the opposite.

> Finding 7. *Coverage degradation does not pose a major concern in choosing hardness dimensions, as it does not necessarily correlate with the conformance improvement brought by the dimension.*

5 DISCUSSION

Our evaluation shows that none of the straightforward metric designs is satisfactory: none sits on or near the approximate Pareto frontier, and all exhibit significant cross-index conformance variation. Still, our results raise more questions and opportunities for future research than they answer, and we discuss some of the implications and future directions below.

Exact and Tight Pareto Frontier. Our greedy search yields a useful, computationally tractable reference, but it remains an approximation in two distinct senses. Being close to it does not necessarily mean a metric is close to a true optimal metric. Furthermore, even if the exact frontier is obtained, it is the upper bound achievable by *any* ordering of the datasets, including orderings that no computable feature of the data could ever reproduce. However, a formal definition of a tight frontier that is realizable by some computable metric is still missing, and how much it differs from our approximation remains unknown. Quantifying this difference would tell us how much of the remaining headroom is worth chasing.

Pushing the Frontier with Workload Adaptation. Given the tradeoff between conformance and coverage, no single metric is best; a set of metrics along the Pareto frontier lets users pick the one that fits their priorities. Yet this tradeoff is inherent only for a workload-agnostic metric that must serve all indexes and datasets at once. As Findings 3 and 4 point out, different learned indexes favor different hardness indicators, so a metric adapted to the target workload can be both more accurate and more applicable than any universal one. Specializing to a user's target indexes thus pushes the frontier outward, rather than merely selecting a point on it. This calls for a workload-aware metric selection framework, taking target indexes as input, weighting the conformance objective toward them, and returning a specialized metric.

Filling Gaps in the Hardness Space. A hardness metric is ultimately valuable not only for ranking existing datasets but for revealing where evaluation is lacking. Consider GRE and CD·PLA-32, the two best two-dimensional metrics. Figures 2 and 6 show how they

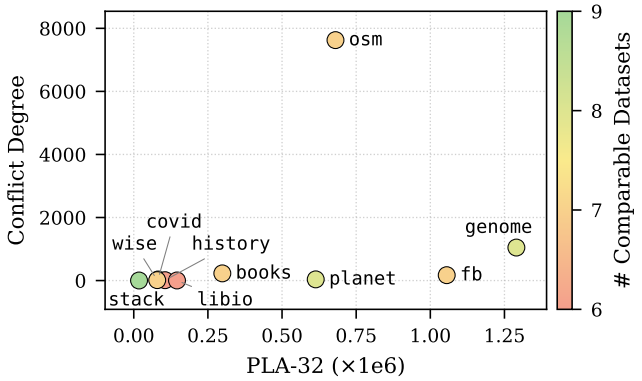

Figure 6: CD·PLA-32 hardness of different datasets, with data point color mapped to the number of comparable datasets each dataset has.

Table 3: Performance and hardness differences between reference real-world datasets and generated datasets that target the GRE hardness of the reference datasets. The hardness difference is reported only for local hardness, as there is no difference in global hardness. The closer to zero the differences are, the better.

Reference Dataset	Performance Diff. (%)		Hardness Diff. (%)
	ALEX	LIPP	
books	−34.23	35.78	−0.36
fb	63.84	6.96	−6.50
osm	5.62	64.32	−2.54
covid	−26.65	−7.10	−0.05
genome	12.91	18.88	−12.23
history	−27.83	−21.91	−0.10
libio	−18.67	−39.24	−0.11
planet	−8.88	0.50	−1.58
stack	−32.10	−59.30	0.00
wise	−33.17	−20.64	−0.03

map the datasets to their respective hardness space. The datasets are visibly clustered in specific regions, leaving gaps in the hardness space where no dataset lies. Since these metrics achieve relatively high conformance, it is important to fill these gaps with more datasets for a more comprehensive evaluation of learned indexes. However, for most metrics we evaluate, there is no straightforward way to generate datasets that meet arbitrary hardness values.

Wongkham et al. [27] have provided an algorithm to generate datasets for their proposed GRE metric. We use it to generate ten datasets with the same GRE values as the ten real-world datasets we evaluate, and evaluate ALEX and LIPP on them.[1] As shown in Table 3, the generated datasets not only fail to match the target hardness values but also lead to different performance characteristics of learned indexes than those of the corresponding real-world

[1]We do not include these datasets in our evaluation as it would introduce a bias and hurt the generality of our findings.

datasets. Therefore, we argue that dataset generation algorithms that reliably realize target hardness values for a wide range of metrics are needed.

6 RELATED WORK

Learned Index Structures. Kraska et al. [9] pioneered a new paradigm of learning-based indexing with the RMI structure: model-based localization followed by a small verification search. After RMI, variants [3, 8] explored lightweight or piecewise approximations and different construction strategies to reduce build cost, space, and lookup latency. The main challenge is *updatability*, where inserts and deletes induce model accuracy drift and structural imbalance. Different designs trade update cost against verification overhead, locality, and range-scan efficiency. Representative updatable learned indexes include ALEX [1], which adopts a top-down design based on gapped arrays and supports adaptive splits and expansions; PGM-index [2], which organizes ϵ-bounded piecewise linear models in a bottom-up manner and handles updates through buffering and merging; and LIPP [29], which maintains the index with an emphasis on precise position prediction. More recent indexes further improve robustness under mixed workloads, particularly for range queries and tail latency [34].

Beyond general-purpose settings, learned indexes have also been specialized for concurrency, hardware, and workload semantics. For multi-core read-write workloads, XIndex and FINEdex are designed to scale under concurrent operations with different update/maintenance mechanisms [11, 24]; SALI further incorporates probability-model-based refinements and adaptive node evolution for scalable concurrent inserts [4]. For sliding-window streams with implicit expirations, SWIX [13] proposes a memory-efficient learned index that supports continuous inserts. There are also complementary directions such as distribution-driven modeling (e.g., DILI [12]), persistent-memory-oriented learned indexes (e.g., APEX [17]), and string-key learned indexes (e.g., SIndex [26]).

Benchmarking Learned Indexes. There have been several recent efforts to benchmark learned indexes. However, none have systematically examined the impact of dataset diversity and workload dynamism. Kipf et al. [7] primarily focus on evaluating static in-memory learned indexes. They use a set of manually chosen real-world datasets for evaluation, comparing learned indexes with traditional structures. Wongkham et al. [27] conducted the first systematic evaluation of updatable learned indexes with a focus on data size, workload, and concurrency. They introduce a data hardness metric in an attempt to capture the distributional hardness of datasets. However, as our analysis has shown, it does not correlate well with index performance. Sun et al. [23] provide a more detailed evaluation of the basic operations of learned indexes, such as lookup, insert, delete, and bulk loading. They additionally cover string keys. Luo et al. [18] focus on robustness, i.e., whether learned indexes can consistently outperform certain traditional indexes across different datasets, sampling methods, bulk loading sizes, and insert patterns.

Several studies have benchmarked learned indexes beyond in-memory scalar-value settings. Lan et al. [10] port multiple updatable learned indexes to disk-resident DBMS settings and perform a comprehensive comparison with B+-trees under diverse workloads. Raman et al. [20] focus on how data sortedness affects the performance of learned indexes versus LSM-trees. They point out that learned indexes may exhibit more unpredictable behavior under different data sortedness. Liu et al. [15] provide an empirical evaluation of multi-dimensional learned indexes.

7 CONCLUSION

We studied the problem of quantifying dataset hardness for learned indexes, and proposed *conformance* and *coverage* as two criteria for evaluating how well a hardness metric reflects index performance and how broadly it applies. Evaluating the GRE metric and several other straightforward metrics under these criteria, we found none fully satisfactory. Still, our evaluation yields useful findings, including effective hardness indicators and guidelines for designing better metrics. We then discussed several open problems and outlined future directions toward better hardness metrics and a more comprehensive evaluation of learned indexes.

ACKNOWLEDGMENTS

We thank Liang Liang, Yuchen Ouyang, and Eleni Zapridou for their valuable input during the early stages of this work. This work was supported by the Swiss State Secretariat for Education, Research and Innovation (SERI) in the Prodasys project, approved by the European Research Council (ERC) as an Advanced Grant.

REFERENCES

[1] Jialin Ding, Umar Farooq Minhas, Jia Yu, Chi Wang, Jaeyoung Do, Yinan Li, Hantian Zhang, Badrish Chandramouli, Johannes Gehrke, Donald Kossmann, David Lomet, and Tim Kraska. 2020. ALEX: An Updatable Adaptive Learned Index. In *Proceedings of the 2020 ACM SIGMOD International Conference on Management of Data* (Portland, OR, USA) *(SIGMOD '20)*. Association for Computing Machinery, New York, NY, USA, 969–984. https://doi.org/10.1145/3318464.3389711

[2] Paolo Ferragina and Giorgio Vinciguerra. 2020. The PGM-index: a fully-dynamic compressed learned index with provable worst-case bounds. *Proc. VLDB Endow.* 13, 8 (April 2020), 1162–1175. https://doi.org/10.14778/3389133.3389135

[3] Alex Galakatos, Michael Markovitch, Carsten Binnig, Rodrigo Fonseca, and Tim Kraska. 2019. FITing-Tree: A Data-aware Index Structure. In *Proceedings of the 2019 International Conference on Management of Data* (Amsterdam, Netherlands) *(SIGMOD '19)*. Association for Computing Machinery, New York, NY, USA, 1189–1206. https://doi.org/10.1145/3299869.3319860

[4] Jiake Ge, Huanchen Zhang, Boyu Shi, Yuanhui Luo, Yunda Guo, Yunpeng Chai, Yuxing Chen, and Anqun Pan. 2023. SALI: A Scalable Adaptive Learned Index Framework based on Probability Models. *Proc. ACM Manag. Data* 1, 4, Article 258 (Dec. 2023), 25 pages. https://doi.org/10.1145/3626752

[5] Google Cloud. 2017. OpenStreetMap. https://console.cloud.google.com/marketplace/details/openstreetmap/geo-openstreetmap

[6] Ali Hadian and Thomas Heinis. 2020. MADEX: Learning-augmented Algorithmic Index Structures. In *AIDB@VLDB 2020, 2nd International Workshop on Applied AI for Database Systems and Applications, Held with VLDB 2020, Monday, August 31, 2020, Online Event / Tokyo, Japan*, Bingsheng He, Berthold Reinwald, and Yingjun Wu (Eds.). https://drive.google.com/file/d/10_CrbAuF8MZQND4gn-DccA2m8QezDQUt/view?usp=sharing

[7] Andreas Kipf, Ryan Marcus, Alexander van Renen, Mihail Stoian, Alfons Kemper, Tim Kraska, and Thomas Neumann. 2019. SOSD: A Benchmark for Learned Indexes. *NeurIPS Workshop on Machine Learning for Systems* (2019).

[8] Andreas Kipf, Ryan Marcus, Alexander van Renen, Mihail Stoian, Alfons Kemper, Tim Kraska, and Thomas Neumann. 2020. RadixSpline: a single-pass learned index. In *Proceedings of the Third International Workshop on Exploiting Artificial Intelligence Techniques for Data Management* (Portland, Oregon) *(aiDM '20)*. Association for Computing Machinery, New York, NY, USA, Article 5, 5 pages. https://doi.org/10.1145/3401071.3401659

[9] Tim Kraska, Alex Beutel, Ed H. Chi, Jeffrey Dean, and Neoklis Polyzotis. 2018. The Case for Learned Index Structures. In *Proceedings of the 2018 International Conference on Management of Data* (Houston, TX, USA) *(SIGMOD '18)*. Association for Computing Machinery, New York, NY, USA, 489–504. https://doi.org/10.1145/3183713.3196909

[10] Hai Lan, Zhifeng Bao, J. Shane Culpepper, and Renata Borovica-Gajic. 2023. Updatable Learned Indexes Meet Disk-Resident DBMS - From Evaluations to Design Choices. *Proc. ACM Manag. Data* 1, 2, Article 139 (June 2023), 22 pages. https://doi.org/10.1145/3589284

[11] Pengfei Li, Yu Hua, Jingnan Jia, and Pengfei Zuo. 2021. FINEdex: a fine-grained learned index scheme for scalable and concurrent memory systems. *Proc. VLDB Endow.* 15, 2 (Oct. 2021), 321–334. https://doi.org/10.14778/3489496.3489512

[12] Pengfei Li, Hua Lu, Rong Zhu, Bolin Ding, Long Yang, and Gang Pan. 2023. DILI: A Distribution-Driven Learned Index. *Proc. VLDB Endow.* 16, 9 (May 2023), 2212–2224. https://doi.org/10.14778/3598581.3598593

[13] Liang Liang, Guang Yang, Ali Hadian, Luis Alberto Croquevielle, and Thomas Heinis. 2024. SWIX: A Memory-efficient Sliding Window Learned Index. *Proc. ACM Manag. Data* 2, 1, Article 41 (March 2024), 26 pages. https://doi.org/10.1145/3639296

[14] Libraries.io. 2017. Repository ID. https://libraries.io/data

[15] Qiyu Liu, Maocheng Li, Yuxiang Zeng, Yanyan Shen, and Lei Chen. 2025. How good are multi-dimensional learned indexes? An experimental survey. *The VLDB Journal* 34, 2 (2025), 17.

[16] Christian E Lopez and Caleb Gallemore. 2021. An augmented multilingual Twitter dataset for studying the COVID-19 infodemic. *Social Network Analysis and Mining* 11, 1 (2021), 102.

[17] Baotong Lu, Jialin Ding, Eric Lo, Umar Farooq Minhas, and Tianzheng Wang. 2021. APEX: a high-performance learned index on persistent memory. *Proc. VLDB Endow.* 15, 3 (Nov. 2021), 597–610. https://doi.org/10.14778/3494124.3494141

[18] Yuanhui Luo, Minhui Xie, Yiheng Tong, Shichao Jiang, and Yunpeng Chai. 2025. Understanding Robustness Issues of Updatable Learned Indexes: [Experiments & Analysis]. *Proc. ACM Manag. Data* 3, 4, Article 270 (Sept. 2025), 25 pages. https://doi.org/10.1145/3749188

[19] Joseph O'Rourke. 1981. An on-line algorithm for fitting straight lines between data ranges. *Commun. ACM* 24, 9 (Sept. 1981), 574–578. https://doi.org/10.1145/358746.358758

[20] Aneesh Raman, Andy Huynh, Jinqi Lu, and Manos Athanassoulis. 2024. Benchmarking learned and lsm indexes for data sortedness. In *Proceedings of the Tenth International Workshop on Testing Database Systems*. 16–22.

[21] Suhas SP Rao, Miriam H Huntley, Neva C Durand, Elena K Stamenova, Ivan D Bochkov, James T Robinson, Adrian L Sanborn, Ido Machol, Arina D Omer, Eric S Lander, et al. 2014. A 3D map of the human genome at kilobase resolution reveals principles of chromatin looping. *Cell* 159, 7 (2014), 1665–1680.

[22] Stackoverflow. 2021. Vote ID. https://archive.org/download/stackexchange

[23] Zhaoyan Sun, Xuanhe Zhou, and Guoliang Li. 2023. Learned Index: A Comprehensive Experimental Evaluation. *Proc. VLDB Endow.* 16, 8 (April 2023), 1992–2004. https://doi.org/10.14778/3594512.3594528

[24] Chuzhe Tang, Youyun Wang, Zhiyuan Dong, Gansen Hu, Zhaoguo Wang, Minjie Wang, and Haibo Chen. 2020. XIndex: a scalable learned index for multicore data storage. In *Proceedings of the 25th ACM SIGPLAN Symposium on Principles and Practice of Parallel Programming* (San Diego, California) *(PPoPP '20)*. Association for Computing Machinery, New York, NY, USA, 308–320. https://doi.org/10.1145/3332466.3374547

[25] Hui Wang, Xin Wang, Jiake Ge, Yunpeng Chai, Lei Liang, and Peng Yi. 2025. High Performance or Low Memory? An Updatable Learned Index Framework for Time-Space Tradeoff. *Proc. ACM Manag. Data* 3, 6, Article 335 (Dec. 2025), 26 pages. https://doi.org/10.1145/3769800

[26] Youyun Wang, Chuzhe Tang, Zhaoguo Wang, and Haibo Chen. 2020. SIndex: a scalable learned index for string keys. In *Proceedings of the 11th ACM SIGOPS Asia-Pacific Workshop on Systems* (Tsukuba, Japan) *(APSys '20)*. Association for Computing Machinery, New York, NY, USA, 17–24. https://doi.org/10.1145/3409963.3410496

[27] Chaichon Wongkham, Baotong Lu, Chris Liu, Zhicong Zhong, Eric Lo, and Tianzheng Wang. 2022. Are updatable learned indexes ready? *Proc. VLDB Endow.* 15, 11 (July 2022), 3004–3017. https://doi.org/10.14778/3551793.3551848

[28] Edward L Wright, Peter RM Eisenhardt, Amy K Mainzer, Michael E Ressler, Roc M Cutri, Thomas Jarrett, J Davy Kirkpatrick, Deborah Padgett, Robert S McMillan, Michael Skrutskie, et al. 2010. The Wide-field Infrared Survey Explorer (WISE): mission description and initial on-orbit performance. *The Astronomical Journal* 140, 6 (2010), 1868–1881.

[29] Jiacheng Wu, Yong Zhang, Shimin Chen, Jin Wang, Yu Chen, and Chunxiao Xing. 2021. Updatable learned index with precise positions. *Proc. VLDB Endow.* 14, 8 (April 2021), 1276–1288. https://doi.org/10.14778/3457390.3457393

[30] Shangyu Wu, Yufei Cui, Jinghuan Yu, Xuan Sun, Tei-Wei Kuo, and Chun Jason Xue. 2022. NFL: robust learned index via distribution transformation. *Proc. VLDB Endow.* 15, 10 (June 2022), 2188–2200. https://doi.org/10.14778/3547305.3547322

[31] Jin Yang, Heejin Yoon, Gyeongchan Yun, Sam H. Noh, and Young-ri Choi. 2023. DyTIS: A Dynamic Dataset Targeted Index Structure Simultaneously Efficient for Search, Insert, and Scan. In *Proceedings of the Eighteenth European Conference on Computer Systems* (Rome, Italy) *(EuroSys '23)*. Association for Computing Machinery, New York, NY, USA, 800–816. https://doi.org/10.1145/3552326.3587434

[32] Jiaoyi Zhang and Yihan Gao. 2022. CARMI: a cache-aware learned index with a cost-based construction algorithm. *Proc. VLDB Endow.* 15, 11 (July 2022), 2679–2691. https://doi.org/10.14778/3551793.3551823

[33] Shunkang Zhang, Ji Qi, Xin Yao, and André Brinkmann. 2024. Hyper: A High-Performance and Memory-Efficient Learned Index via Hybrid Construction. *Proc. ACM Manag. Data* 2, 3, Article 145 (May 2024), 26 pages. https://doi.org/10.1145/3654948

[34] Xinyi Zhang, Liang Liang, Anastasia Ailamaki, and Jianliang Xu. 2025. HIRE: A Hybrid Learned Index for Robust and Efficient Performance under Mixed Workloads. arXiv:2511.21307 [cs.DB] https://arxiv.org/abs/2511.21307