Learned Indexes From the One-dimensional to the Multi-dimensional Spaces: Challenges, Techniques, and Opportunities

Abdullah Al-Mamun
Purdue University
West Lafayette, USA
mamuna@purdue.edu

Jianguo Wang
Purdue University
West Lafayette, USA
csjgwang@purdue.edu

Walid G. Aref
Purdue University
West Lafayette, USA
aref@purdue.edu

Abstract

Recently, the class of learned index structures has emerged as one form of database indexes that make use of Machine Learning (ML) techniques. The learned indexes designed for the one-dimensional space have demonstrated improvements in both the query processing time and the index size. Observing the advantages of one-dimensional learned indexes, various learned indexes have been proposed for the multi-dimensional space. This class of learned indexes is termed "Learned Multi-dimensional Indexes." This tutorial on learned indexes is designed based on our long survey article on the subject [4]. In this tutorial, we use a taxonomy to categorize over 100 learned one- and multi-dimensional indexes with more focus on the class of learned multi-dimensional indexes. The goal of this tutorial is to explain the fundamental techniques behind state-of-the-art learned one- and multi-dimensional indexes with emphasis on the latter, and identify the ongoing challenges and future opportunities for research in this domain.

CCS Concepts

• **Information systems → Data management systems**.

Keywords

Machine Learning for Database Systems, Learned Index, Learned Multi-dimensional Index

ACM Reference Format:
Abdullah Al-Mamun, Jianguo Wang, and Walid G. Aref. 2025. Learned Indexes From the One-dimensional to the Multi-dimensional Spaces: Challenges, Techniques, and Opportunities. In *Companion of the 2025 International Conference on Management of Data (SIGMOD-Companion '25), June 22–27, 2025, Berlin, Germany*. ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3722212.3725639

1 Introduction

Machine Learning (ML) techniques have been successfully applied to enhance the performance of core database systems components, e.g., indexes [59] and query optimizers [84]. Particularly, treating a database index structure as an ML model is a key observation in building *learned* index structures. Initially, learned indexes have been designed for the one-dimensional case [27, 35, 59]. Due to the performance benefits of one-dimensional learned indexes, the concept has been extended for the multi-dimensional space [58]. Numerous learned multi-dimensional indexes have been proposed in the literature, e.g., [3, 4, 24, 28, 63, 66, 90, 96]. Learned index structures can be broadly categorized into pure and hybrid learned indexes. Pure learned indexes are primarily designed to replace traditional index structures, e.g., to replace the B-tree [9, 17], the R-tree [42], or the Bloom filter [13]. On the other hand, hybrid learned indexes combine ML models with traditional indexes to build an ML-enhanced index structure. The spectrum of learned indexes is given in Figure 1.

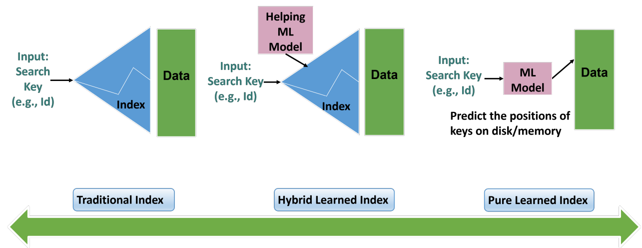

Figure 1: Spectrum of learned index structures.

In this tutorial, we use a taxonomy that we have developed in our survey paper [4] (also highlighted in Figure 2) to classify over 100 learned one- and multi-dimensional indexes. The concept of learned indexes has been extended into the multi-dimensional space due to the promising performance of the learned one-dimensional indexes. The following four approaches have been adopted to design various learned multi-dimensional indexes:

- Approach 1: Traditional multi-dimensional indexes are augmented with ML models.
- Approach 2: Multi-dimensional data are projected into the one-dimensional space and a learned one-dimensional index is built in the projected space.
- Approach 3: A learned one-dimensional index is applied once per dimension, on one or more dimensions of the multi-dimensional space without using any projection functions.
- Approach 4: Learned multi-dimensional indexes are designed from scratch by leveraging a suitably arranged data layout.

The various learned one-dimensional indexes have influenced the design choices for learned multi-dimensional indexes. As a result, in this tutorial, an overview of learned indexes for the one-dimensional space will be covered first, and this will serve as the foundation for learned indexes for the multi-dimensional space.

SIGMOD-Companion '25, Berlin, Germany

ACM ISBN 979-8-4007-1564-8/2025/06
https://doi.org/10.1145/3722212.3725639

In the taxonomy, in the context of learning the index, we refer to the task of learning the mapping from keys to positions inside a data set as the learned index. First, we differentiate between **Immutable vs. Mutable learned indexes** based on the capability of supporting dynamic inserts/updates. We further classify the mutable learned indexes based on **Fixed vs. Dynamic data layouts**. If the data layout is fixed before the construction of an index, we refer to the index as having a fixed data layout. In contrast, in the context of dynamic data layouts, ML models help re-arrange the data layout during the index construction phase.

Next, the learned indexes are categorized based on the underlying data space: **One-dimensional vs. Multi-dimensional space**. Both the learned one- and multi-dimensional indexes are categorized to identify their position in the spectrum (Figure 1) of learned indexes: **Pure vs. Hybrid**. In the context of mutable pure learned indexes, we also distinguish the indexes based on their adopted strategies for supporting new data insertions: **In-place vs. Delta buffer insertion strategies**. The hybrid learned indexes are categorized based on their traditional index component (e.g., B-tree, R-tree, etc.). Finally, the taxonomy sub-categorizes further the class of learned multi-dimensional indexes based on how they deal with the underlying space into **Projected vs. Native space**. For example, some learned multi-dimensional indexes use Space Filling Curves (SFC, for short) [88, 104] to map the multi-dimensional space into the one-dimensional space, and then apply a learned model to learn the mapped data in the one-dimensional space.

The evolution of research related to learned indexes is presented in Figure 3. It is evident from Figure 3 that the field of learned one- and multi-dimensional indexes is evolving rapidly. As a result, this tutorial will cover the current state of research in this highly active field with more focus on learned indexes for the multi-dimensional case, and present several open challenges for future research.

2 Tutorial Outline

We plan to deliver a 1.5-hour lecture-style tutorial. Moreover, we split the tutorial into the following parts:

- Introduction and Historical Background (10 Minutes)
  - Learning the Index vs. Indexing the Learned Models
  - Evolution of Learned Indexes
- Part 1: Learned Indexes for the One-dimensional Space (20 minutes)
  - Immutable vs. Mutable Indexes (5 minutes)
  - Fixed vs. Dynamic Data Layouts (5 minutes)
  - Pure vs. Hybrid Indexes (5 minutes)
  - In-place vs. Delta Buffer Insertion Strategies (5 minutes)
- Part 2: Learned Indexes for the Multi-dimensional Space (55 minutes)
  - Motivation and Challenges (5 minutes)
    - Projected vs. Native Space
  - Learned Multi-dimensional Indexes (50 minutes)
    - Immutable Indexes (10 minutes)
      - Immutable Pure Indexes (5 minutes)
      - Immutable Hybrid Indexes (5 minutes)
    - Mutable Indexes with Fixed Data Layouts (20 minutes)
      - Pure Indexes with In-place vs. Delta Buffer Insertion Strategies (10 minutes)
      - Hybrid Indexes (10 minutes)
    - Mutable Indexes with Dynamic Data Layouts (20 minutes)
      - Pure Indexes with In-place vs. Delta Buffer Insertion Strategies (10 minutes)
      - Hybrid Indexes (10 minutes)
- Part 3: Open Challenges for Future Research (5 minutes)

3 Historical Background

3.1 Learning the Index vs. Indexing the Learned Models

In this tutorial, we first differentiate between the concepts of **Indexing the Learned Models vs. Learning the Index**. In the context of indexing the learned models, e.g., [7], the focus is to index the learned (i.e., trained) ML models so that we can distinguish among these models in an efficient way during query processing. On the other hand, in the context of learning the index, we refer to the task of learning the mapping (i.e., learned index) from keys to positions inside a data set. We will give some historical background about several early approaches, e.g., [52, 107] for indexing the learned models to contrast them against the more recent trend of learning the index.

3.2 Evolution of Learned Indexes

We will present and discuss the evolution of research for both one- and multi-dimensional learned indexes using Figure 3. The Figure 3 is adapted from our survey paper on learned indexes [4]. Here, we have grouped the papers on learned one- and multi-dimensional indexes based on their publication years. Moreover, we have used the → symbol to indicate if a later paper is related to an earlier one. The ⇢ is used when there is a line crossing. Additionally, a ∘ symbol is used to denote a copy of an earlier paper in a later year. One- and multi-dimensional indexes are differentiated using the □ and △ symbols, respectively.

4 Part 1: Learned Indexes for the One-dimensional Space

The Recursive Model Index (RMI, for short) [59] is considered the first instance of a "Learned Index" for the one-dimensional space. In [59], the key insight is that traditional indexes (e.g., the B-tree) can be perceived as an ML (i.e., predictive) models. For example, given an input search key, a B-tree predicts (i.e., searches) the key's position in a logical sorted array. Hence, an ML model can potentially serve as a replacement for a traditional index. RMI proposes the idea of learning the Cumulative Distribution Function (CDF) to learn the key-to-position mapping. However, learning this mapping is challenging, and might not be accurate. As a result, an error correction mechanism is used after each ML model prediction. Notice that a key assumption is that the data is totally ordered (i.e., sorted) in the one-dimensional space. After the introduction of RMI, various indexes have been proposed in the literature addressing different issues related to the learned indexes: supporting dynamic inserts/updates, e.g., [27, 35, 129], supporting concurrency, e.g., [116]. Moreover, in the context of hybrid learned indexes, ML-enhanced

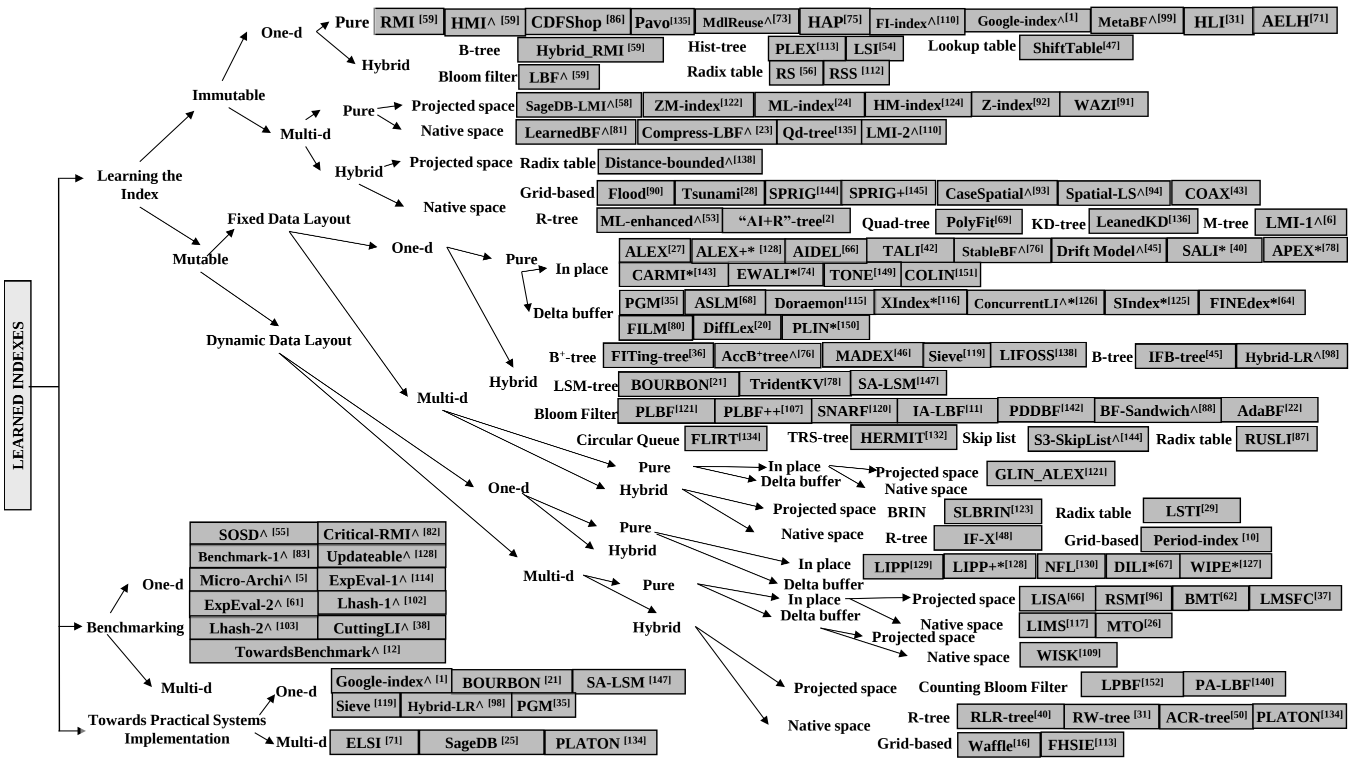

Figure 2: Taxonomy of learned indexes. A wedge (∧) symbol indicates that we have assigned a name to refer a particular paper or an index whenever a fixed name is not given by the original authors of that paper. An asterisk (*) symbol is used if the index natively supports concurrency. The hybrid learned indexes are categorized based on their underlying traditional data structures. The end of a branch indicates that there are no papers in that category as of the time this article has been written.

variants of traditional index structures, e.g., Bloom filter [13], LSM-tree [79], Skiplist [95]) have been introduced [21, 87, 143].

In Part 1, we will introduce the taxonomy, and present the fundamental concepts of learned index structures. Moreover, we will cover the existing learned indexes designed for the one-dimensional space. Particularly, we will follow the taxonomy, and explain the core concepts of a representative learned one-dimensional index from each class of indexes. Part 1 will serve as the foundation for Part 2 of the tutorial.

4.1 Immutable vs. Mutable Indexes

In the taxonomy (Figure 2), in the context of learning the index, we will first differentiate between immutable and mutable learned indexes based on the capability of supporting dynamic inserts/updates. The class of *Immutable* one-dimensional indexes contains 18 indexes [1, 19, 30, 47, 54, 56, 59, 70, 72, 74, 85, 98, 108, 111, 112, 132]. On the other hand, in the taxonomy, the class of *Mutable* one-dimensional indexes contains 48 indexes [11, 20–22, 27, 35, 36, 39, 41, 44–46, 64, 65, 67, 68, 73, 75–78, 80, 86, 87, 97, 106, 115, 116, 118–120, 125–128, 128–131, 133, 137, 141–143, 146, 148–150]. We will present and discuss the core ideas of representative immutable and mutable one-dimensional learned indexes.

4.2 Fixed vs. Dynamic Data Layout

The class of mutable learned indexes are further classified into two categories based on the data layout during the index construction phase. We will present the core ideas of representative mutable one-dimensional learned indexes (e.g., BOURBON [21], LIPP [129]) to explain the concepts of fixed and dynamic data layouts.

4.3 Pure vs. Hybrid Indexes

In the taxonomy, both immutable and mutable learned indexes are categorized into two types: Pure vs. Hybrid (see Figure 1). The class of immutable learned one-dimensional indexes contains 11 pure and 7 hybrid indexes. For example, we will cover RMI [59], and Hybrid_RMI [59] as the representative indexes to illustrate the pure and hybrid categories, respectively.

4.4 In-place vs. Delta Buffer Insertion Strategy

The class of mutable pure indexes are further categorized into two types based on the strategy for supporting new data insertions. For example, we will present the ALEX [27], and the PGM [35] as the representative mutable pure indexes to explain the concepts of In-place and Delta Buffer insertion strategies, respectively.

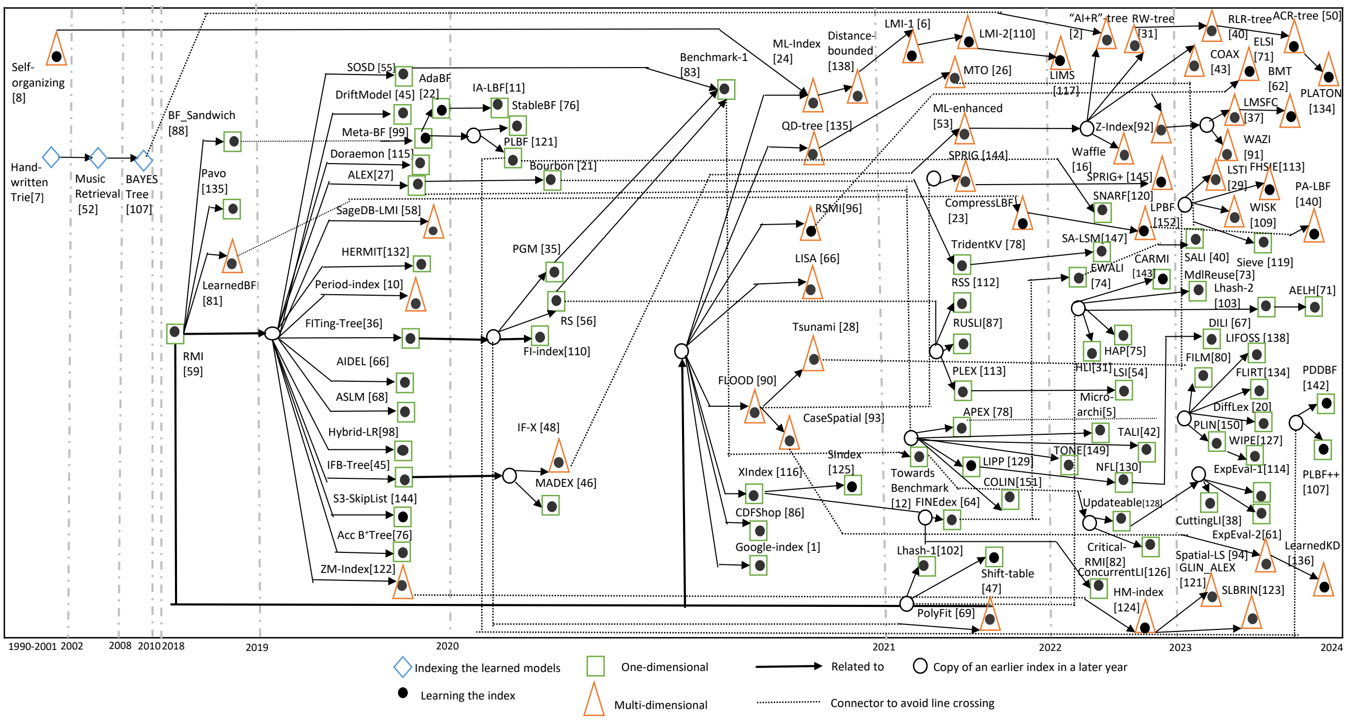

Figure 3: Evolution of learned indexes. Lines connecting between the various learned indexes reflect dependence of these indexes on earlier work.

5 Part 2: Learned indexes for the Multi-dimensional space

In Part 2, we will cover the state-of-the-art by presenting the motivation and challenges, and explaining the core concepts of the learned multi-dimensional indexes. Moreover, we will use the taxonomy to group similar indexes together, and highlight the similarities and differences within each group. Notice that we will also highlight the open branches in the taxonomy during our presentation.

5.1 Motivation and Challenges

Although the term learned multi-dimensional indexes has been introduced recently, one of the earliest papers on a distribution-aware index structure for spatial (i.e., multi-dimensional) data can be found in [8]. We will begin Part 2 by presenting the motivation behind extending the concept of learned one-dimensional indexes into the multi-dimensional space. We will discuss the additional challenges related to the development of learned multi-dimensional indexes. Moreover, we will also introduce the taxonomy categories that are only applicable to the learned multi-dimensional indexes: Projected vs. Native Space.

5.2 Immutable Pure Indexes

The class of *Immutable Pure* learned indexes, e.g., [23, 24, 58, 81, 91, 92, 110, 122, 124, 135], operate on the projected space by using a Space-filling Curve order (e.g., Z-order [89], Hilbert order [49]). Some indexes in this class, e.g., the ZM-index [122] and Qd-tree [135], operate on the original representation of the data (i.e, native space). We will explain the core concepts behind this class of learned indexes.

5.3 Immutable Hybrid Indexes

The class of *Immutable Hybrid* learned indexes, e.g., [2, 6, 28, 43, 53, 69, 90, 93, 94, 136, 138, 144, 145], is classified based on the underling traditional index component used (e.g., R-tree [42], Quad-tree [105]). We will present representative indexes from each group (e.g., grid-based, tree-based) of hybrid indexes.

5.4 Mutable Indexes with Fixed Data Layout

In the taxonomy, there are several indexes [10, 29, 48, 121, 123] in this category of mutable indexes with fixed data layout. Moreover, the pure indexes with fixed data layout are further categorized into two branches based on their new data insertion strategy: In-place vs. Delta Buffer. On the other hand, the hybrid indexes are further categorized based on their underlying traditional index structures. We will present the core ideas and the similarities/differences of the learned indexes in each of the branches.

5.5 Mutable Indexes with Dynamic Data Layout

Indexes in this category [16, 26, 31, 37, 40, 50, 62, 66, 96, 109, 113, 117, 134, 140, 152] are classified in the category of mutable indexes with dynamic data layout. Moreover, the pure indexes with dynamic data layout are further categorized into two branches based on their new data insertion strategy: In-place vs. Delta Buffer. On the other hand, the hybrid indexes are further categorized based on their underlying traditional index structures. We will cover the representative learned multi-dimensional indexes from each of the branches according to the taxonomy.

5.6 A Summary of ML Techniques, Supported Query Types, and Practical Systems Integration

We will present a summary of the various ML techniques used for learned one- and multi-dimensional indexes. Moreover, a summary of the supported query types (e.g., point, range, kNN, and join queries) for each of the 40+ learned multi-dimensional indexes will be included. On the other hand, several learned one-dimensional indexes have been successfully integrated into practical systems [1, 21, 35, 97, 118, 146]. We will also discuss similar efforts in the context of learned multi-dimensional indexes [25, 71, 134].

6 Part 3: Open Challenges and Future Directions

In Part 3, we will discuss several open challenges and future research opportunities related to learned one- and multi-dimensional indexes. The open challenges and directions for future research are as follows:

6.1 The Lack of Total Ordering and Error Bound for the Multi-dimensional Case

Due to the lack of total ordering in the multi-dimensional space, it is challenging to provide an error bound in case of ML model mispredictions. As a result, a class of learned multi-dimensional indexes projects multi-dimensional data into the one-dimensional space to impose an ordering. However, some learned multi-dimensional indexes might select one of the dimensions to impose an order in native space. Although various methods has been proposed to address this challenge, there is still room for improvement.

6.2 Choice of ML models

In the context of learned indexes, to reduce the impact of ML model training time, storage overhead, and prediction latency, simple ML models are adopted whenever possible. However, choosing an ML model for a particular learned index design is a challenging task. As a result, a learned index should avoid using complex ML models so that the model building time and prediction latency do not become bottlenecks for achieving low index construction time and high query processing performance, respectively.

6.3 ML Model Re-training

In the context of learned indexes, changes in the underlying input data/query distribution should be detected as soon as possible, and a model re-training process should be triggered when necessary. For example, exploring the concept of Machine Unlearning [60] in the context of mutable learned indexes is an interesting research direction

6.4 Supporting Dynamic Inserts/Updates

Although mutable learned one- and multi-dimensional indexes can support inserts/updates, further research is needed to investigate the advantages and disadvantages of each approach, e.g., as in [14].

6.5 Supporting Concurrency

Only a few proposed methods (marked with an * symbol in the taxonomy Figure 2) discuss the issue of concurrency in the context of learned indexes. As a result, future research in this area should treat the issue of concurrency as a first-class citizen while designing learned one- and multi-dimensional indexes.

6.6 Index Compression

Both one- and multi-dimensional indexes have demonstrated significant benefits in terms of reduced storage requirements. Although the advantages of a learned multi-dimensional bloom filter for index compression are studied in CompressLBF [23], exploring the potential benefits of index compression using learned one- and multi-dimensional indexes in the context of other index types is an interesting direction for future research.

6.7 Security

The issue of security in the context of learned indexes is little explored. The impact of poisoning attacks has been discussed in a recent study [57]. Particularly, indexes, e.g., PGM [35], that are designed with a worst-case guarantee are expected to perform well in the presence of adversarial queries. Exploring this direction in the context of one- and multi-dimensional learned indexes is an open research topic.

6.8 Benchmarking for the Multi-dimensional Case

There are extensive benchmarking studies for various learned one dimensional indexes [5, 12, 38, 55, 61, 82, 83, 102, 103, 114, 128]. However, similar effort is still missing in the context of learned multi-dimensional indexes.

6.9 Theoretical Analysis for the Multi-dimensional Case

There are a few theoretical studies [15, 32–34, 139] that mathematically analyze the reasons behind the performance gain of learned one-dimensional indexes over traditional indexes. Theoretical analysis of the various components of learned multi-dimensional indexes is needed to better understand the benefits and limitations of existing techniques.

6.10 Leveraging Modern Hardware for Learned Indexes

The benefit of natively implementing a learned one-dimensional index on a GPU has been presented in [151]. Similar investigation in the context of other learned one- and multi-dimensional indexes is an interesting direction for future research.

7 Related Tutorials

There is a prior tutorial on the subject of learned multi-dimensional indexes [3] that has been offered by the same authors of this tutorial.[1] However, the prior tutorial [3] covers methods only till the end of the year 2020. Hence, this tutorial serves as a significant extension over that earlier tutorial [3] by using a more sophisticated taxonomy to cover the newer varieties of learned multi-dimensional indexes. A survey paper on learned indexes by the same authors of this tutorial can be found in [4]. Another related tutorial can be found in [51]. The tutorial [51] presents the initial results related to learned indexes. However, in [51], only the Flood [90] index structure was presented as the representative learned multi-dimensional index. In a recent tutorial on spatial query optimization [147], seven learned spatial (i.e., multi-dimensional) indexes have been included in the context of I/O cost estimation. Another recent tutorial on machine learning for databases [18] covers only the ML-enhanced variants of R-tree index. In contrast, in addition to covering learned one-dimensional indexes as background, this tutorial will provide a comprehensive overview of learned multi-dimensional indexes by including over 40 learned multi-dimensional indexes. A series of related tutorials on big spatial data can be found in [99–101]. However, the focus of these series of tutorials is not on learned index structures.

8 Target Audiences and Goals

Intended Audience. This tutorial is intended for a broad category of students, academics, researchers, and practitioners with basic knowledge of data structures, algorithms, and machine learning. The tutorial is designed to be self-contained in providing all the necessary background. We only assume basic understanding of fundamental data indexing structures (e.g., the B-tree, the R-Tree, and the Bloom Filter).

Learning Goals. The target outcomes include:

- Understanding the fundamental concepts of learned index structures for the one- and multi-dimensional spaces.
- Familiarity of state-of-the-art learned one-dimensional indexes and thorough coverage of the state-of-the-art of learned multi-dimensional indexes using a taxonomy.
- Knowing the evolution of learned indexes using a timeline figure and recognizing their relationships and differences.
- Identifying the open challenges and future research opportunities in the area of learned one- and multi-dimensional indexes.

9 Short Biographies

Abdullah Al-Mamun is a PhD candidate at the Department of Computer Science, Purdue University. His research interests are in the area of Machine Learning for Database Systems, particularly, in the field of Learned Multi- and High-dimensional Index Structures.

Jianguo Wang is a tenure-track assistant professor at the Department of Computer Science, Purdue University. He received his Ph.D. degree in Computer Science from UC San Diego. His research interests include disaggregated databases and vector databases. His research has won multiple prestigious awards, including the ACM SIGMOD Research Highlight Award, the NSF CAREER Award, and the IEEE TCDE Rising Star Award.

Walid G. Aref is a professor at the Department of Computer Science, Purdue University. His research interests are in extending the functionality of database systems in support of emerging applications, e.g., spatial, spatio-temporal and graph databases. He is also interested in query processing, indexing, and data streaming. He has served as the Editor-in-Chief of the ACM Transactions of Spatial Algorithms and Systems (ACM TSAS), an editor of the VLDB Journal and the ACM Transactions of Database Systems (ACM TODS). Walid has won several best paper awards including the 2016 VLDB ten-year best paper award. He is a Fellow of the IEEE, and a member of the ACM.

[1] https://www.cs.purdue.edu/homes/aref/learned-indexes-tutorial.html

Acknowledgments

Walid G. Aref and Jianguo Wang acknowledge the support of the National Science Foundation under Grant Numbers IIS-1910216 and IIS-2337806.

References

[1] Hussam Abu-Libdeh, Deniz Altınbüken, Alex Beutel, Ed H Chi, Lyric Doshi, Tim Kraska, Andy Ly, Christopher Olston, et al. 2020. Learned indexes for a google-scale disk-based database. *arXiv preprint arXiv:2012.12501* (2020).
[2] Abdullah Al-Mamun, Ch Md Rakin Haider, Jianguo Wang, and Walid G Aref. 2022. The "AI+ R"-tree: An Instance-optimized R-tree. In *2022 23rd IEEE International Conference on Mobile Data Management (MDM)*. IEEE, 9–18.
[3] Abdullah Al-Mamun, Hao Wu, and Walid G Aref. 2020. A Tutorial on Learned Multi-dimensional Indexes. In *Proceedings of the 28th International Conference on Advances in Geographic Information Systems*. 1–4.
[4] Abdullah Al-Mamun, Hao Wu, Qiyang He, Jianguo Wang, and Walid G. Aref. 2024. A Survey of Learned Indexes for the Multi-dimensional Space. arXiv:2403.06456
[5] Mikkel Møller Andersen and Pinar Tözün. 2022. Micro-architectural analysis of a learned index. In *Proceedings of the Fifth International Workshop on Exploiting Artificial Intelligence Techniques for Data Management*. 1–12.
[6] Matej Antol, Jaroslav Ol'ha, Terézia Slanináková, and Vlastislav Dohnal. 2021. Learned metric index—proposition of learned indexing for unstructured data. *Information Systems* 100 (2021), 101774.
[7] Walid Aref, Daniel Barbará, and Padmavathi Vallabhaneni. 1995. The Handwritten Trie: Indexing Electronic Ink. *SIGMOD Rec.* 24, 2 (May 1995), 151–162.
[8] G Phanendra Babu. 1997. Self-organizing neural networks for spatial data. *Pattern Recognition Letters* 18, 2 (1997), 133–142.
[9] Rudolf Bayer and Edward McCreight. 1970. Organization and maintenance of large ordered indices. In *Proceedings of the 1970 ACM SIGFIDET (Now SIGMOD) Workshop on Data Description, Access and Control*. 107–141.
[10] Andreas Behrend, Anton Dignös, Johann Gamper, Philip Schmiegelt, Hannes Voigt, Matthias Rottmann, and Karsten Kahl. 2019. Period Index: A Learned 2D Hash Index for Range and Duration Queries. In *Proceedings of the 16th International Symposium on Spatial and Temporal Databases*. 100–109.
[11] Arindam Bhattacharya, Srikanta Bedathur, and Amitabha Bagchi. 2020. Adaptive learned bloom filters under incremental workloads. In *Proceedings of the 7th ACM IKDD CoDS and 25th COMAD*. 107–115.
[12] Laurent Bindschaedler, Andreas Kipf, Tim Kraska, Ryan Marcus, and Umar Farooq Minhas. 2021. Towards a Benchmark for Learned Systems. In *2021 IEEE 37th International Conference on Data Engineering Workshops (ICDEW)*. IEEE, 127–133.
[13] Burton H Bloom. 1970. Space/time trade-offs in hash coding with allowable errors. *Commun. ACM* 13, 7 (1970), 422–426.
[14] SUBARNA CHATTERJEE, MARK F PEKALA, LEV KRUGLYAK, and STRATOS IDREOS. 2024. Limousine: Blending Learned and Classical Indexes to Self-Design Larger-than-Memory Cloud Storage Engines. (2024).
[15] Daoyuan Chen, Wuchao Li, Yaliang Li, Bolin Ding, Kai Zeng, Defu Lian, and Jingren Zhou. 2022. Learned Index with Dynamic ϵ. In *The Eleventh International Conference on Learning Representations*.
[16] Dalsu Choi, Hyunsik Yoon, Hyubjin Lee, and Yon Dohn Chung. 2022. Waffle: in-memory grid index for moving objects with reinforcement learning-based configuration tuning system. *Proceedings of the VLDB Endowment* 15, 11 (2022), 2375–2388.
[17] Douglas Comer. 1979. Ubiquitous B-tree. *ACM Computing Surveys (CSUR)* 11, 2 (1979), 121–137.

[18] Gao Cong, Jingyi Yang, and Yue Zhao. 2024. Machine learning for databases: Foundations, paradigms, and open problems. In *Companion of the 2024 International Conference on Management of Data*. 622–629.

[19] Andrew Crotty. 2021. Hist-Tree: Those Who Ignore It Are Doomed to Learn.. In *CIDR*.

[20] Lixiao Cui, Kedi Yang, Yusen Li, Gang Wang, and Xiaoguang Liu. 2023. DiffLex: A High-Performance, Memory-Efficient and NUMA-Aware Learned Index using Differentiated Management. In *Proceedings of the 52nd International Conference on Parallel Processing*. 62–71.

[21] Yifan Dai, Yien Xu, Aishwarya Ganesan, Ramnatthan Alagappan, Brian Kroth, Andrea Arpaci-Dusseau, and Remzi Arpaci-Dusseau. 2020. From {WiscKey} to Bourbon: A Learned Index for {Log-Structured} Merge Trees. In *14th USENIX Symposium on Operating Systems Design and Implementation (OSDI 20)*. 155–171.

[22] Zhenwei Dai and Anshumali Shrivastava. 2019. Adaptive learned Bloom filter (Ada-BF): Efficient utilization of the classifier. *arXiv preprint arXiv:1910.09131* (2019).

[23] Angjela Davitkova, Damjan Gjurovski, and Sebastian Michel. 2021. Compressing (Multidimensional) Learned Bloom Filters. In *Workshop on Databases and AI*.

[24] Angjela Davitkova, Evica Milchevski, and Sebastian Michel. 2020. The ML-Index: A Multidimensional, Learned Index for Point, Range, and Nearest-Neighbor Queries.. In *EDBT*. 407–410.

[25] Jialin Ding, Ryan Marcus, Andreas Kipf, Vikram Nathan, Aniruddha Nrusimha, Kapil Vaidya, Alexander van Renen, and Tim Kraska. 2022. SageDB: An Instance-Optimized Data Analytics System. *Proceedings of the VLDB Endowment* 15, 13 (2022), 4062–4078.

[26] Jialin Ding, Umar Farooq Minhas, Badrish Chandramouli, Chi Wang, Yinan Li, Ying Li, Donald Kossmann, Johannes Gehrke, and Tim Kraska. 2021. Instance-Optimized Data Layouts for Cloud Analytics Workloads. In *SIGMOD Conference*. ACM, 418–431.

[27] Jialin Ding, Umar Farooq Minhas, Jia Yu, Chi Wang, Jaeyoung Do, Yinan Li, Hantian Zhang, Badrish Chandramouli, Johannes Gehrke, Donald Kossmann, et al. 2020. ALEX: an updatable adaptive learned index. In *Proceedings of the 2020 ACM SIGMOD International Conference on Management of Data*. 969–984.

[28] Jialin Ding, Vikram Nathan, Mohammad Alizadeh, and Tim Kraska. 2020. Tsunami: A Learned Multi-dimensional Index for Correlated Data and Skewed Workloads. *Proc. VLDB Endow.* 14, 2 (2020), 74–86.

[29] Xiaofeng Ding, Yinting Zheng, Zuan Wang, Kim-Kwang Raymond Choo, and Hai Jin. 2023. A learned spatial textual index for efficient keyword queries. *Journal of Intelligent Information Systems* 60, 3 (2023), 803–827.

[30] Yuquan Ding, Xujian Zhao, and Peiquan Jin. 2022. An Error-Bounded Space-Efficient Hybrid Learned Index with High Lookup Performance. In *International Conference on Database and Expert Systems Applications*. Springer, 216–228.

[31] Haowen Dong, Chengliang Chai, Yuyu Luo, Jiabin Liu, Jianhua Feng, and Chaoqun Zhan. 2022. RW-Tree: A Learned Workload-aware Framework for R-tree Construction. In *2022 IEEE 38th International Conference on Data Engineering (ICDE)*. IEEE, 2073–2085.

[32] Paolo Ferragina, Fabrizio Lillo, and Giorgio Vinciguerra. 2020. Why are learned indexes so effective?. In *International Conference on Machine Learning*. PMLR, 3123–3132.

[33] Paolo Ferragina, Fabrizio Lillo, and Giorgio Vinciguerra. 2021. On the performance of learned data structures. *Theoretical Computer Science* 871 (2021), 107–120.

[34] Paolo Ferragina and Giorgio Vinciguerra. 2020. Learned data structures. In *Recent Trends in Learning From Data*. Springer, 5–41.

[35] Paolo Ferragina and Giorgio Vinciguerra. 2020. The PGM-index: a fully-dynamic compressed learned index with provable worst-case bounds. *Proceedings of the VLDB Endowment* 13, 8 (2020), 1162–1175.

[36] Alex Galakatos, Michael Markovitch, Carsten Binnig, Rodrigo Fonseca, and Tim Kraska. 2019. FITing-Tree: A Data-aware Index Structure. In *SIGMOD Conference*. ACM, 1189–1206.

[37] Jian Gao, Xin Cao, Xin Yao, Gong Zhang, and Wei Wang. 2023. LMSFC: A Novel Multidimensional Index Based on Learned Monotonic Space Filling Curves. *Proc. VLDB Endow.* 16, 10 (aug 2023), 2605–2617.

[38] Jiake Ge, Boyu Shi, Yanfeng Chai, Yuanhui Luo, Yunda Guo, Yinxuan He, and Yunpeng Chai. 2023. Cutting Learned Index into Pieces: An In-depth Inquiry into Updatable Learned Indexes. In *2023 IEEE 39th International Conference on Data Engineering (ICDE)*. IEEE, 315–327.

[39] 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. *arXiv preprint arXiv:2308.15012* (2023).

[40] Tu Gu, Kaiyu Feng, Gao Cong, Cheng Long, Zheng Wang, and Sheng Wang. 2023. The RLR-Tree: A Reinforcement Learning Based R-Tree for Spatial Data. *Proceedings of the ACM on Management of Data* 1, 1 (2023), 1–26.

[41] Na Guo, Yaqi Wang, Haonan Jiang, Xiufeng Xia, and Yu Gu. 2022. TALI: An Update-Distribution-Aware Learned Index for Social Media Data. *Mathematics* 10, 23 (2022), 4507.

[42] Antonin Guttman. 1984. R-trees: A dynamic index structure for spatial searching. In *Proceedings of the 1984 ACM SIGMOD international conference on Management of data*. 47–57.

[43] Ali Hadian, Behzad Ghaffari, Taiyi Wang, and Thomas Heinis. 2023. COAX: Correlation-Aware Indexing. In *2023 IEEE 39th International Conference on Data Engineering Workshops (ICDEW)*. IEEE, 55–59.

[44] Ali Hadian and Thomas Heinis. 2019. Considerations for handling updates in learned index structures. In *Proceedings of the Second International Workshop on Exploiting Artificial Intelligence Techniques for Data Management*. ACM, 3.

[45] Ali Hadian and Thomas Heinis. 2019. Interpolation-friendly B-trees: Bridging the Gap Between Algorithmic and LearnedIndexes. (2019).

[46] Ali Hadian and Thomas Heinis. 2020. Madex: Learning-augmented algorithmic index structures. In *Proceedings of the 2nd International Workshop on Applied AI for Database Systems and Applications*.

[47] Ali Hadian and Thomas Heinis. 2021. Shift-Table: A low-latency learned index for range queries using model correction. *arXiv preprint arXiv:2101.10457* (2021).

[48] Ali Hadian, Ankit Kumar, and Thomas Heinis. 2020. Hands-off Model Integration in Spatial Index Structures. In *Proceedings of the 2nd International Workshop on Applied AI for Database Systems and Applications*.

[49] David Hilbert. 1891. Ueber die stetige Abbildung einer Linie auf ein Flächenstück. *Math. Ann.* 38 (1891), 459–460.

[50] Shuai Huang, Yong Wang, and Guoliang Li. 2023. ACR-Tree: Constructing R-Trees Using Deep Reinforcement Learning. In *International Conference on Database Systems for Advanced Applications*. Springer, 80–96.

[51] Stratos Idreos and Tim Kraska. 2019. From Auto-tuning One Size Fits All to Self-designed and Learned Data-intensive Systems (Tutorial). In *Proceedings of the 2019 International Conference on Management of Data*. 2054–2059. http://people.csail.mit.edu/kraska/pub/sigmod19tutorialpart2.pdf

[52] Hui Jin and HV Jagadish. 2002. Indexing Hidden Markov Models for Music Retrieval.. In *ISMIR*.

[53] Rong Kang, Wentao Wu, Chen Wang, Ce Zhang, and Jianmin Wang. 2021. The Case for ML-Enhanced High-Dimensional Indexes. In *Proceedings of the 3rd International Workshop on Applied AI for Database Systems and Applications*.

[54] Andreas Kipf, Dominik Horn, Pascal Pfeil, Ryan Marcus, and Tim Kraska. 2022. Lsi: A learned secondary index structure. In *Proceedings of the Fifth International Workshop on Exploiting Artificial Intelligence Techniques for Data Management*. 1–5.

[55] Andreas Kipf, Ryan Marcus, Alexander van Renen, Mihail Stoian, Alfons Kemper, Tim Kraska, and Thomas Neumann. 2019. SOSD: A Benchmark for Learned Indexes. *ArXiv* abs/1911.13014 (2019).

[56] 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*. 1–5.

[57] Evgenios M Kornaropoulos, Silei Ren, and Roberto Tamassia. 2022. The price of tailoring the index to your data: Poisoning attacks on learned index structures. In *Proceedings of the 2022 International Conference on Management of Data*. 1331–1344.

[58] Tim Kraska, Mohammad Alizadeh, Alex Beutel, Ed H Chi, Jialin Ding, Ani Kristo, Guillaume Leclerc, Samuel Madden, Hongzi Mao, and Vikram Nathan. 2019. Sagedb: A learned database system. (2019).

[59] 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*. ACM, 489–504.

[60] Meghdad Kurmanji, Eleni Triantafillou, and Peter Triantafillou. 2024. Machine unlearning in learned databases: An experimental analysis. *Proceedings of the ACM on Management of Data* 2, 1 (2024), 1–26.

[61] Hai Lan, Zhifeng Bao, J Shane Culpepper, and Renata Borovica-Gajic. 2023. Updatable Learned Indexes Meet Disk-Resident DBMS-From Evaluations to Design Choices. *Proceedings of the ACM on Management of Data* 1, 2 (2023), 1–22.

[62] Jiangneng Li, Zheng Wang, Gao Cong, Cheng Long, Han Mao Kiah, and Bin Cui. 2023. Towards Designing and Learning Piecewise Space-Filling Curves. *Proceedings of the VLDB Endowment* 16, 9 (2023), 2158–2171.

[63] Mingxin Li, Hancheng Wang, Haipeng Dai, Meng Li, Rong Gu, Feng Chen, Zhiyuan Chen, Shuaituan Li, Qizhi Liu, and Guihai Chen. 2024. A Survey of Multi-Dimensional Indexes: Past and Future Trends. *IEEE Transactions on Knowledge and Data Engineering* (2024).

[64] Pengfei Li, Yu Hua, Jingnan Jia, and Pengfei Zuo. 2021. FINEdex: a fine-grained learned index scheme for scalable and concurrent memory systems. *Proceedings of the VLDB Endowment* 15, 2 (2021), 321–334.

[65] Pengfei Li, Yu Hua, Pengfei Zuo, and Jingnan Jia. 2019. A Scalable Learned Index Scheme in Storage Systems. *CoRR* abs/1905.06256 (2019). arXiv:1905.06256

[66] Pengfei Li, Hua Lu, Qian Zheng, Long Yang, and Gang Pan. 2020. LISA: A Learned Index Structure for Spatial Data. *SIGMOD* (2020).

[67] Pengfei Li, Hua Lu, Rong Zhu, Bolin Ding, Long Yang, and Gang Pan. 2023. DILI: A Distribution-Driven Learned Index. 16, 9 (2023), 2212–2224.

[68] Xin Li, Jingdong Li, and Xiaoling Wang. 2019. ASLM: Adaptive single layer model for learned index. In *International Conference on Database Systems for Advanced Applications*. Springer, 80–95.

[69] Zhe Li, Tsz Nam Chan, Man Lung Yiu, and Christian S Jensen. 2021. PolyFit: Polynomial-based indexing approach for fast approximate range aggregate queries. In *Advances in Database Technology-24th International Conference on Extending Database Technology, EDBT 2021*. OpenProceedings. org, 241–252.

[70] Yuming Lin, Zhengguo Huang, and You Li. 2023. Learning hash index based on a shallow autoencoder. *Applied Intelligence* 53, 12 (2023), 14999–15010.

[71] Guanli Liu, Jianzhong Qi, Christian S. Jensen, James Bailey, and Lars Kulik. 2023. Efficiently Learning Spatial Indices. In *2023 IEEE 39th International Conference on Data Engineering (ICDE)*. 1572–1584.

[72] Guanli Liu, Jianzhong Qi, Lars Kulik, Kazuya Soga, Renata Borovica-Gajic, and Benjamin IP Rubinstein. 2023. Efficient Index Learning via Model Reuse and Fine-tuning. In *2023 IEEE 39th International Conference on Data Engineering Workshops (ICDEW)*. IEEE, 60–66.

[73] Li Liu, Chunhua Li, Zhou Zhang, Yuhan Liu, Ke Zhou, and Ji Zhang. 2022. A Data-aware Learned Index Scheme for Efficient Writes. In *Proceedings of the 51st International Conference on Parallel Processing*. 1–11.

[74] Qiyu Liu, Yanyan Shen, and Lei Chen. 2022. HAP: an efficient hamming space index based on augmented pigeonhole principle. In *Proceedings of the 2022 International Conference on Management of Data*. 917–930.

[75] Qiyu Liu, Libin Zheng, Yanyan Shen, and Lei Chen. 2020. Stable learned bloom filters for data streams. *Proceedings of the VLDB Endowment* 13, 12 (2020), 2355–2367.

[76] A Llavesh, Utku Sirin, R West, and A Ailamaki. 2019. Accelerating b+ tree search by using simple machine learning techniques. In *Proceedings of the 1st International Workshop on Applied AI for Database Systems and Applications*.

[77] Baotong Lu, Jialin Ding, Eric Lo, Umar Farooq Minhas, and Tianzheng Wang. 2021. APEX: a high-performance learned index on persistent memory. *Proceedings of the VLDB Endowment* 15, 3 (2021), 597–610.

[78] Kai Lu, Nannan Zhao, Jiguang Wan, Changhong Fei, Wei Zhao, and Tongliang Deng. 2021. TridentKV: A Read-Optimized LSM-Tree Based KV Store via Adaptive Indexing and Space-Efficient Partitioning. *IEEE Transactions on Parallel and Distributed Systems* 33, 8 (2021), 1953–1966.

[79] Chen Luo and Michael J Carey. 2020. LSM-based storage techniques: a survey. *The VLDB Journal* 29, 1 (2020), 393–418.

[80] Chaohong Ma, Xiaohui Yu, Yifan Li, Xiaofeng Meng, and Aishan Maoliniyazi. 2022. Film: A fully learned index for larger-than-memory databases. *Proceedings of the VLDB Endowment* 16, 3 (2022), 561–573.

[81] Stephen Macke, Alex Beutel, Tim Kraska, Maheswaran Sathiamoorthy, Derek Zhiyuan Cheng, and Ed H Chi. 2018. Lifting the curse of multidimensional data with learned existence indexes. In *Workshop on ML for Systems at NeurIPS*. 1–6.

[82] Marcel Maltry and Jens Dittrich. 2022. A critical analysis of recursive model indexes. *Proceedings of the VLDB Endowment* 15, 5 (2022), 1079–1091.

[83] Ryan Marcus, Andreas Kipf, Alexander van Renen, Mihail Stoian, Sanchit Misra, Alfons Kemper, Thomas Neumann, and Tim Kraska. 2020. Benchmarking learned indexes. *Proceedings of the VLDB Endowment* 14, 1 (2020), 1–13.

[84] Ryan Marcus, Parimarjan Negi, Hongzi Mao, Chi Zhang, Mohammad Alizadeh, Tim Kraska, Olga Papaemmanouil, and Nesime Tatbul. 2019. Neo: a learned query optimizer. *Proceedings of the VLDB Endowment* 12, 11 (2019), 1705–1718.

[85] Ryan Marcus, Emily Zhang, and Tim Kraska. 2020. CDFShop: Exploring and Optimizing Learned Index Structures. *SIGMOD* (2020).

[86] Mayank Mishra and Rekha Singhal. 2021. RUSLI: real-time updatable spline learned index. In *Fourth Workshop in Exploiting AI Techniques for Data Management*. 1–8.

[87] Michael Mitzenmacher. 2018. A Model for Learned Bloom Filters, and Optimizing by Sandwiching. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems* (Montréal, Canada) *(NIPS'18)*. Curran Associates Inc., Red Hook, NY, USA, 462–471.

[88] Mohamed F Mokbel, Walid G Aref, and Ibrahim Kamel. 2003. Analysis of multi-dimensional space-filling curves. *GeoInformatica* 7 (2003), 179–209.

[89] Guy M Morton. 1966. A computer oriented geodetic data base and a new technique in file sequencing. (1966).

[90] Vikram Nathan, Jialin Ding, Mohammad Alizadeh, and Tim Kraska. 2020. Learning multi-dimensional indexes. In *Proceedings of the 2020 ACM SIGMOD International Conference on Management of Data*. 985–1000.

[91] Sachith Pai, Michael Mathioudakis, and Yanhao Wang. 2023. Workload-aware and Learned Z-Indexes. *arXiv preprint arXiv:2310.04268* (2023).

[92] Sachith Gopalakrishna Pai, Michael Mathioudakis, and Yanhao Wang. 2022. Towards an Instance-Optimal Z-Index. *AIDB@VLDB* (2022).

[93] Varun Pandey, Alexander van Renen, Andreas Kipf, Ibrahim Sabek, Jialin Ding, and Alfons Kemper. 2020. The Case for Learned Spatial Indexes. *arXiv preprint arXiv:2008.10349* (2020).

[94] Varun Pandey, Alexander van Renen, Eleni Tzirita Zacharatou, Andreas Kipf, Ibrahim Sabek, Jialin Ding, Volker Markl, and Alfons Kemper. 2023. Enhancing In-Memory Spatial Indexing with Learned Search. *arXiv preprint arXiv:2309.06354* (2023).

[95] William Pugh. 1990. Skip lists: a probabilistic alternative to balanced trees. *Commun. ACM* 33, 6 (1990), 668–676.

[96] Jianzhong Qi, Guanli Liu, Christian S Jensen, and Lars Kulik. 2020. Effectively learning spatial indices. *Proceedings of the VLDB Endowment* 13, 12 (2020), 2341–2354.

[97] Wenwen Qu, Xiaoling Wang, Jingdong Li, and Xin Li. 2019. Hybrid indexes by exploring traditional B-tree and linear regression. In *International Conference on Web Information Systems and Applications*. Springer, 601–613.

[98] Jack Rae, Sergey Bartunov, and Timothy Lillicrap. 2019. Meta-learning neural bloom filters. In *International Conference on Machine Learning*. PMLR, 5271–5280.

[99] Ibrahim Sabek and Mohamed F Mokbel. 2019. Machine learning meets big spatial data. *Proceedings of the VLDB Endowment* 12, 12 (2019), 1982–1985.

[100] Ibrahim Sabek and Mohamed F Mokbel. 2020. Machine learning meets big spatial data. In *2020 IEEE 36th International Conference on Data Engineering (ICDE)*. IEEE, 1782–1785.

[101] Ibrahim Sabek and Mohamed F Mokbel. 2021. Machine learning meets big spatial data (revised). In *2021 22nd IEEE International Conference on Mobile Data Management (MDM)*. IEEE, 5–8.

[102] Ibrahim Sabek, Kapil Vaidya, Dominik Horn, Andreas Kipf, and Tim Kraska. 2021. When Are Learned Models Better Than Hash Functions? *arXiv preprint arXiv:2107.01464* (2021).

[103] Ibrahim Sabek, Kapil Vaidya, Dominik Horn, Andreas Kipf, Michael Mitzenmacher, and Tim Kraska. 2022. Can Learned Models Replace Hash Functions? *Proceedings of the VLDB Endowment* 16, 3 (2022), 532–545.

[104] Hans Sagan. 2012. *Space-filling curves*. Springer Science & Business Media.

[105] Hanan Samet. 1984. The quadtree and related hierarchical data structures. *ACM Computing Surveys (CSUR)* 16, 2 (1984), 187–260.

[106] Atsuki Sato and Yusuke Matsui. 2023. Fast Partitioned Learned Bloom Filter. *arXiv preprint arXiv:2306.02846* (2023).

[107] Thomas Seidl, Ira Assent, Philipp Kranen, Ralph Krieger, and Jennifer Herrmann. 2009. Indexing density models for incremental learning and anytime classification on data streams. In *Proceedings of the 12th international conference on extending database technology: advances in database technology*. 311–322.

[108] Naufal Fikri Setiawan, Benjamin IP Rubinstein, and Renata Borovica-Gajic. 2020. Function interpolation for learned index structures. In *Databases Theory and Applications: 31st Australasian Database Conference, ADC 2020, Melbourne, VIC, Australia, February 3–7, 2020, Proceedings 31*. Springer, 68–80.

[109] Yufan Sheng, Xin Cao, Yixiang Fang, Kaiqi Zhao, Jianzhong Qi, Gao Cong, and Wenjie Zhang. 2023. WISK: A Workload-aware Learned Index for Spatial Keyword Queries. *Proceedings of the ACM on Management of Data* 1, 2 (2023), 1–27.

[110] Terézia Slanináková, Matej Antol, Jaroslav Oľha, Vojtěch Kaňa, and Vlastislav Dohnal. 2021. Data-driven learned metric index: an unsupervised approach. In *International Conference on Similarity Search and Applications*. Springer, 81–94.

[111] Benjamin Spector, Andreas Kipf, Kapil Vaidya, Chi Wang, Umar Farooq Minhas, and Tim Kraska. 2021. Bounding the last mile: Efficient learned string indexing. *arXiv preprint arXiv:2111.14905* (2021).

[112] Mihail Stoian, Andreas Kipf, Ryan Marcus, and Tim Kraska. 2021. PLEX: towards practical learned indexing. In *3rd International Workshop on Applied AI for Database Systems and Applications*.

[113] Xinyu Su, Jianzhong Qi, and Egemen Tanin. 2023. A Fast Hybrid Spatial Index with External Memory Support. In *2023 IEEE 39th International Conference on Data Engineering Workshops (ICDEW)*. IEEE, 67–73.

[114] Zhaoyan Sun, Xuanhe Zhou, and Guoliang Li. 2023. Learned Index: A Comprehensive Experimental Evaluation. *Proceedings of the VLDB Endowment* 16, 8 (2023), 1992–2004.

[115] Chuzhe Tang, Zhiyuan Dong, Minjie Wang, Zhaoguo Wang, and Haibo Chen. 2019. Learned Indexes for Dynamic Workloads. *arXiv preprint arXiv:1902.00655* (2019).

[116] 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*. 308–320.

[117] Yao Tian, Tingyun Yan, Xi Zhao, Kai Huang, and Xiaofang Zhou. 2022. A learned index for exact similarity search in metric spaces. *IEEE Transactions on Knowledge and Data Engineering* (2022).

[118] Yulai Tong, Jiazhen Liu, Hua Wang, Ke Zhou, Rongfeng He, Qin Zhang, and Cheng Wang. 2023. Sieve: A Learned Data-Skipping Index for Data Analytics. *Proceedings of the VLDB Endowment* 16, 11 (2023), 3214–3226.

[119] Kapil Vaidya, Subarna Chatterjee, Eric Knorr, Michael Mitzenmacher, Stratos Idreos, and Tim Kraska. 2022. SNARF: a learning-enhanced range filter. *Proceedings of the VLDB Endowment* 15, 8 (2022), 1632–1644.

[120] Kapil Vaidya, Eric Knorr, Michael Mitzenmacher, and Tim Kraska. 2020. Partitioned Learned Bloom Filters. In *International Conference on Learning Representations*.

[121] Congying Wang and Jia Yu. 2022. GLIN: A Lightweight Learned Indexing Mechanism for Complex Geometries. *arXiv preprint arXiv:2207.07745* (2022).

[122] Haixin Wang, Xiaoyi Fu, Jianliang Xu, and Hua Lu. 2019. Learned Index for Spatial Queries. In *2019 20th IEEE International Conference on Mobile Data Management (MDM)*. IEEE, 569–574.

[123] Lijun Wang, Linshu Hu, Chenhua Fu, Yuhan Yu, Peng Tang, Feng Zhang, and Renyi Liu. 2023. SLBRIN: A Spatial Learned Index Based on BRIN. *ISPRS International Journal of Geo-Information* 12, 4 (2023), 171.

[124] Ning Wang and Jianqiu Xu. 2021. Spatial queries based on learned index. In *Spatial Data and Intelligence: First International Conference, SpatialDI 2020, Virtual Event, May 8–9, 2020, Proceedings 1*. Springer, 245–257.

[125] 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*. 17–24.

[126] Zhaoguo Wang, Haibo Chen, Youyun Wang, Chuzhe Tang, and Huan Wang. 2022. The concurrent learned indexes for multicore data storage. *ACM Transactions on Storage (TOS)* 18, 1 (2022), 1–35.

[127] Zhonghua Wang, Chen Ding, Fengguang Song, Kai Lu, Jiguang Wan, Zhihu Tan, Changsheng Xie, and Guokuan Li. 2023. WIPE: a Write-Optimized Learned Index for Persistent Memory. *ACM Transactions on Architecture and Code Optimization* (2023).

[128] Chaichon Wongkham, Baotong Lu, Chris Liu, Zhicong Zhong, Eric Lo, and Tianzheng Wang. 2022. Are updatable learned indexes ready? *Proceedings of the VLDB Endowment* 15, 11 (2022), 3004–3017.

[129] 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 (apr 2021), 1276–1288.

[130] Shangyu Wu, Yufei Cui, Jinghuan Yu, Xuan Sun, Tei-Wei Kuo, and Chun Jason Xue. 2022. NFL: robust learned index via distribution transformation. *Proceedings of the VLDB Endowment* 15, 10 (2022), 2188–2200.

[131] Yingjun Wu, Jia Yu, Yuanyuan Tian, Richard Sidle, and Ronald Barber. 2019. Designing Succinct Secondary Indexing Mechanism by Exploiting Column Correlations. *arXiv preprint arXiv:1903.11203* (2019).

[132] Wenkun Xiang, Hao Zhang, Rui Cui, Xing Chu, Keqin Li, and Wei Zhou. 2018. Pavo: A RNN-Based Learned Inverted Index, Supervised or Unsupervised? *IEEE Access* 7 (2018), 293–303.

[133] Guang Yang, Liang Liang, Ali Hadian, and Thomas Heinis. 2023. FLIRT: A Fast Learned Index for Rolling Time frames. (2023).

[134] Jingyi Yang and Gao Cong. 2023. PLATON: Top-down R-tree Packing with Learned Partition Policy. *Proceedings of the ACM on Management of Data* 1, 4 (2023), 1–26.

[135] Zongheng Yang, Badrish Chandramouli, Chi Wang, Johannes Gehrke, Yinan Li, Umar Farooq Minhas, Per-Åke Larson, Donald Kossmann, and Rajeev Acharya. 2020. Qd-tree: Learning Data Layouts for Big Data Analytics. In *SIGMOD Conference*. ACM, 193–208.

[136] Peng Yongxin, Zhou Wei, Zhang Lin, and Du Hongle. 2020. A Study of Learned KD Tree Based on Learned Index. In *2020 International Conference on Networking and Network Applications (NaNA)*. IEEE, 355–360.

[137] Tong Yu, Guanfeng Liu, An Liu, Zhixu Li, and Lei Zhao. 2023. LIFOSS: a learned index scheme for streaming scenarios. *World Wide Web* 26, 1 (2023), 501–518.

[138] Eleni Tzirita Zacharatou, Andreas Kipf, Ibrahim Sabek, Varun Pandey, Harish Doraiswamy, and Volker Markl. 2021. The Case for Distance-Bounded Spatial Approximations. In *CIDR*. www.cidrdb.org.

[139] Sepanta Zeighami and Cyrus Shahabi. 2023. On Distribution Dependent Sub-Logarithmic Query Time of Learned Indexing. *arXiv preprint arXiv:2306.10651* (2023).

[140] Meng Zeng, Beiji Zou, Xiaoyan Kui, Chengzhang Zhu, Ling Xiao, Zhi Chen, Jingyu Du, et al. 2023. PA-LBF: Prefix-Based and Adaptive Learned Bloom Filter for Spatial Data. *International Journal of Intelligent Systems* 2023 (2023).

[141] Meng Zeng, Beiji Zou, Wensheng Zhang, Xuebing Yang, Guilan Kong, Xiaoyan Kui, and Chengzhang Zhu. 2023. Two-layer partitioned and deletable deep bloom filter for large-scale membership query. *Information Systems* 119 (2023), 102267.

[142] Jiaoyi Zhang and Yihan Gao. 2022. CARMI: a cache-aware learned index with a cost-based construction algorithm. *Proceedings of the VLDB Endowment* 15, 11 (2022), 2679–2691.

[143] Jingtian Zhang, Sai Wu, Zeyuan Tan, Gang Chen, Zhushi Cheng, Wei Cao, Yusong Gao, and Xiaojie Feng. 2019. S3: A scalable in-memory skip-list index for key-value store. *Proceedings of the VLDB Endowment* 12, 12 (2019), 2183–2194.

[144] Songnian Zhang, Suprio Ray, Rongxing Lu, and Yandong Zheng. 2021. SPRIG: A Learned Spatial Index for Range and kNN Queries. In *SSTD*. ACM, 96–105.

[145] Songnian Zhang, Suprio Ray, Rongxing Lu, and Yandong Zheng. 2022. Efficient Learned Spatial Index With Interpolation Function Based Learned Model. *IEEE Transactions on Big Data* (2022).

[146] Teng Zhang, Jian Tan, Xin Cai, Jianying Wang, Feifei Li, and Jianling Sun. 2022. SA-LSM: optimize data layout for LSM-tree based storage using survival analysis. *Proceedings of the VLDB Endowment* 15, 10 (2022), 2161–2174.

[147] Xin Zhang and Ahmed Eldawy. 2024. Spatial Query Optimization With Learning. *Proceedings of the VLDB Endowment* 17, 12 (2024), 4245–4248.

[148] Yong Zhang, Xinran Xiong, and Oana Balmau. 2022. TONE: cutting tail-latency in learned indexes. In *Proceedings of the Workshop on Challenges and Opportunities of Efficient and Performant Storage Systems*. 16–23.

[149] Zhou Zhang, Zhaole Chu, Peiquan Jin, Yongping Luo, Xike Xie, Shouhong Wan, Yun Luo, Xufei Wu, Peng Zou, Chunyang Zheng, et al. 2022. Plin: a persistent learned index for non-volatile memory with high performance and instant recovery. *Proceedings of the VLDB Endowment* 16, 2 (2022), 243–255.

[150] Zhou Zhang, Pei-Quan Jin, Xiao-Liang Wang, Yan-Qi Lv, Shou-Hong Wan, and Xi-Ke Xie. 2021. COLIN: a cache-conscious dynamic learned index with high read/write performance. *Journal of Computer Science and Technology* 36 (2021), 721–740.

[151] Xun Zhong, Yong Zhang, Yu Chen, Chao Li, and Chunxiao Xing. 2022. Learned Index on GPU. In *2022 IEEE 38th International Conference on Data Engineering Workshops (ICDEW)*. IEEE, 117–122.

[152] Beiji Zou, Meng Zeng, Chengzhang Zhu, Ling Xiao, and Zhi Chen. 2022. A learned prefix bloom filter for spatial data. In *International Conference on Database and Expert Systems Applications*. Springer, 336–350.