QUERY PROCESSING TRADEOFFS OVER AN ML-ENHANCED R-TREE

Authors: Abdullah Al-Mamun, Ch. Md. Rakin Haider, Walid G. Aref

Purdue University

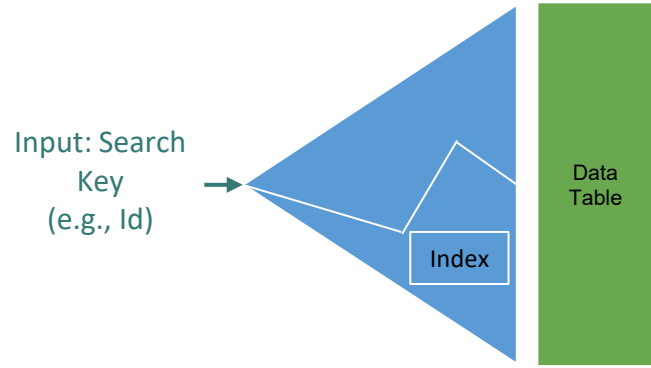
West Lafayette, Indiana, USA

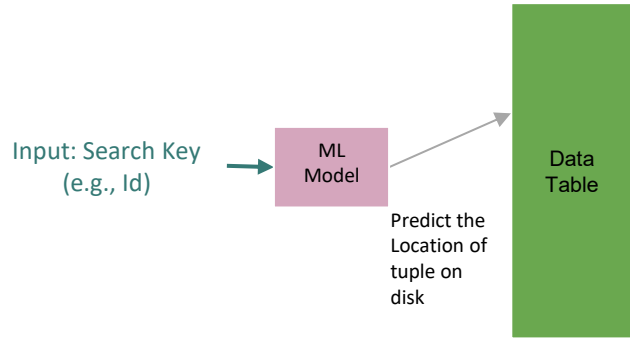
Venue: GeoAI@SIGSPATIAL, Minneapolis, USA

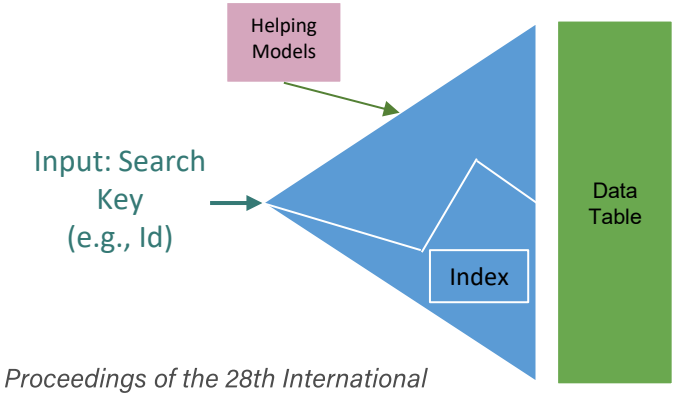
PURDUE UNIVERSITY

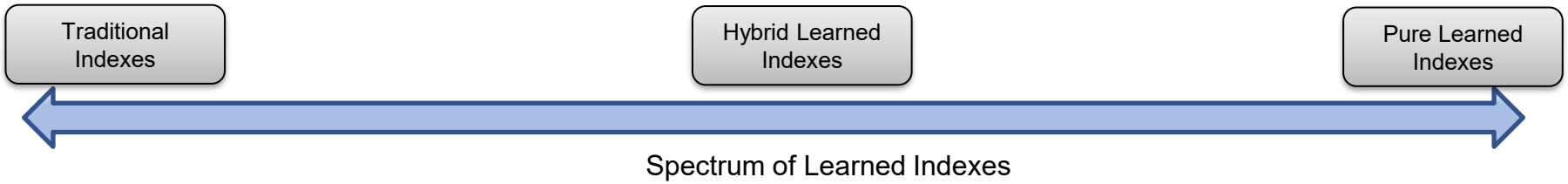
Background: Learned Index Structures

- Traditional Indexes:
  - Theoretical guarantee on performance
  - Well studied and successfully integrated in real systems
- Pure Learned Indexes:
  - Learn search-key distribution with some error correction mechanism
  - Better performance with less space requirement
- Hybrid Learned Indexes:
  - Optimizing traditional indexes with helping (e.g., ML) models





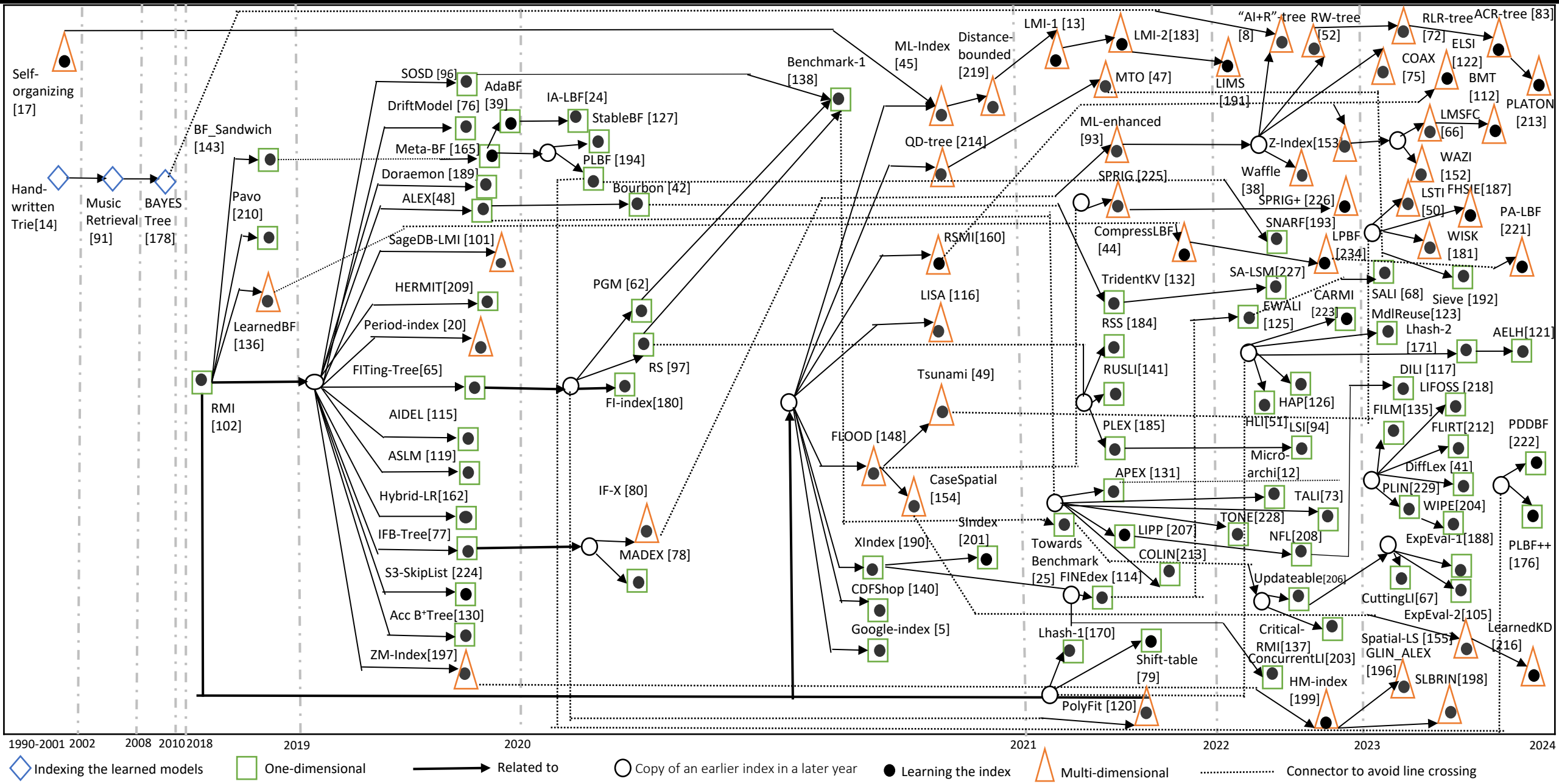




PURDUE UNIVERSITY

Abdullah Al-Mamun, Hao Wu, and Walid G. Aref. "A tutorial on learned multi-dimensional indexes." In *Proceedings of the 28th International Conference on Advances in Geographic Information Systems*, pp. 1-4. 2020.

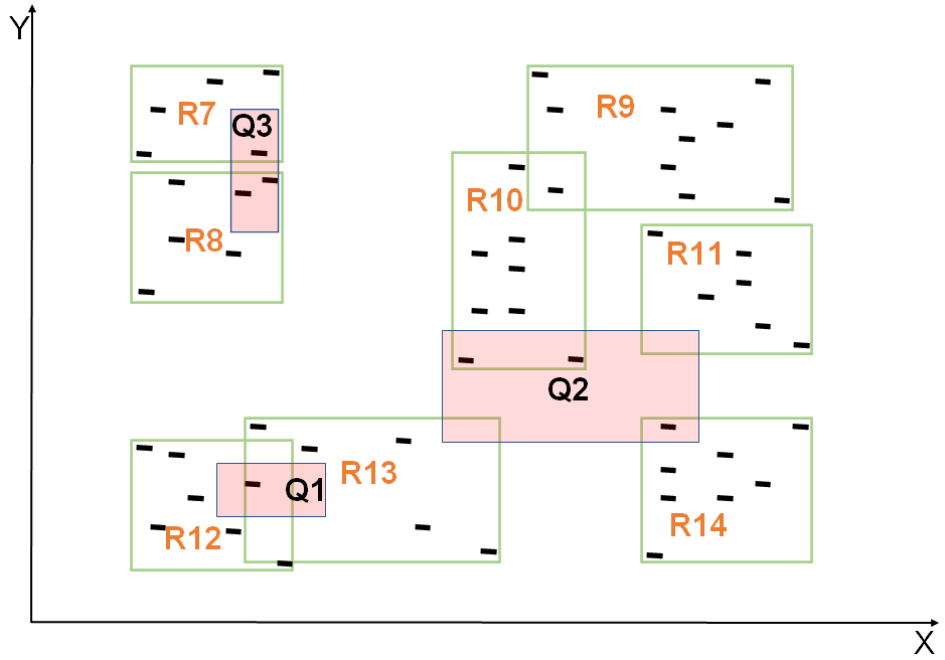
Background: Evolution of Learned Indexes



Abdullah Al-Mamun, Hao Wu, Qiyang He, Jianguo Wang, and Walid G. Aref. "A Survey of Learned Indexes for the Multi-dimensional Space." *ACM Computing Surveys (CSUR), 2025.*

Background: The R-tree

- R-tree is a widely-used spatial index structure.
- For range query processing in R-tree:
  - Even if the bounding box of a leaf node overlaps the query region, the data object itself may not
    - Scan all the data objects contained inside the leaf node
    - Test whether the data object is contained in the given range
      - (or “overlap” the query region for non-zero objects)



Query	Visited leaf nodes	True leaf nodes
Q1	R12, R13	R12
Q2	R10, R11, R13, R14	R10, R14
Q3	R7, R8	R7, R8

PURDUE UNIVERSITY

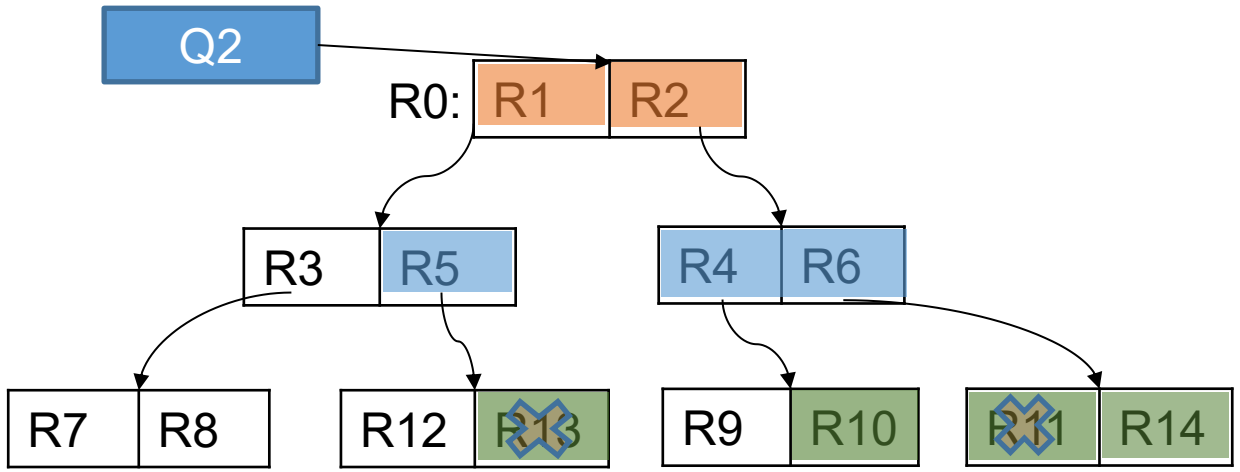
Motivation

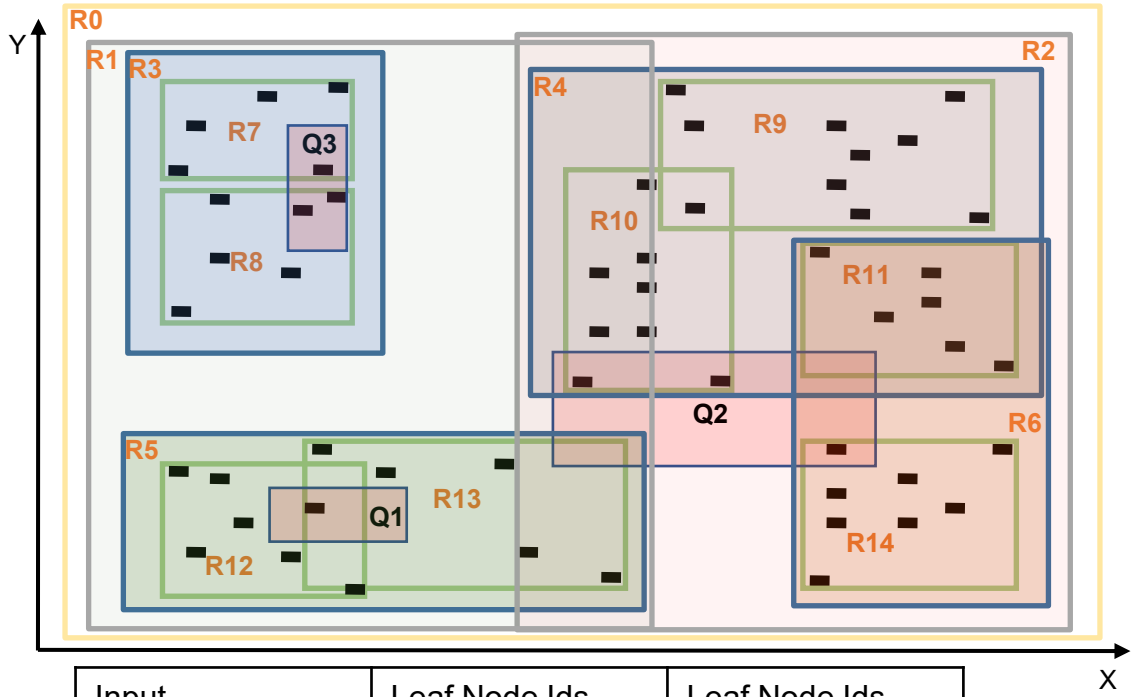
- Given the benefits of learned index structures and the issue of node overlap in the R-tree, the following important questions arise:
  - Which workloads degrade the performance of range query processing in a traditional R-tree?
  - If identified, can we leverage ML techniques for the R-tree to make R-tree query processing faster?
  - How to choose an appropriate ML model?
  - What are the tradeoffs in processing queries over the ML-enhanced R-tree?

PURDUE UNIVERSITY®

Problem Formulation

- Given a range query $Q(X_{min}, Y_{min}, X_{max}, Y_{max})$ as input, can we directly predict the true leafNodeIds in the R-Tree that contain the output data objects?



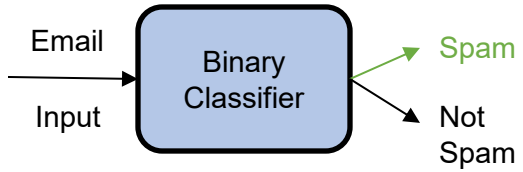


Input	Leaf Node Ids that will be searched by R-Tree	Leaf Node Ids that contains the actual data
Query-1	R12, R13	R13
Query-2	R10, R11, R13, R14	R10, R14

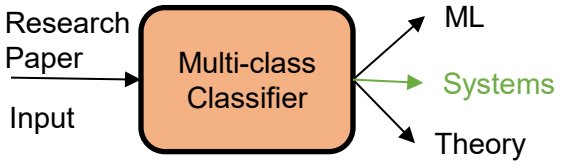
PURDUE UNIVERSITY

R-tree Search Operation: A Multi-label Classification Task

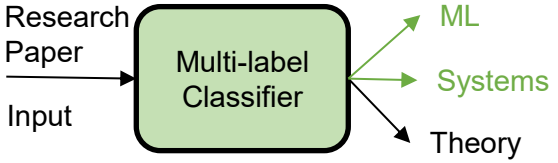
- Three different types of classification tasks:



Total number of classes = 2
For an input, the number of output classes = 1

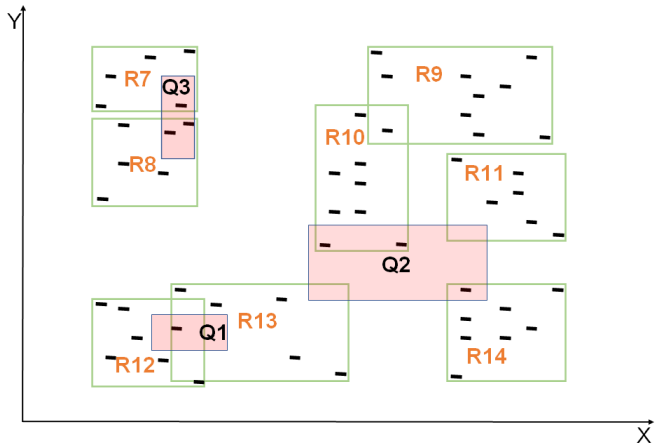


Total number of classes > 2
For an input, the number of output classes = 1

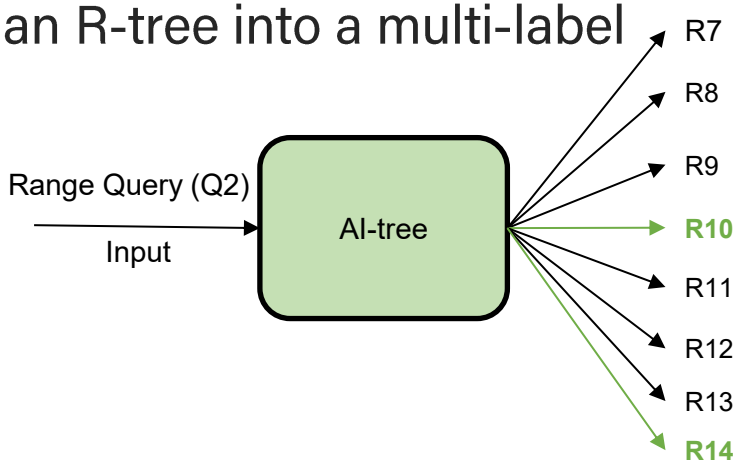


Total number of classes > 2
For an input, the number of output classes ≥ 0

- Proposal:
  - The AI-tree which transforms the search operation of an R-tree into a multi-label classification task.



An example of leaf nodes in an R-tree



Total number of classes > 2
For an input, the number of output classes ≥ 0

PURDUE UNIVERSITY®

Abdullah Al-Mamun, Haider, Ch Md Rakin, Jianguo Wang, and Walid G. Aref. "The "AI+ R"-tree: An Instance-optimized R-tree." In 2022 23rd IEEE International Conference on Mobile Data Management (MDM), pp. 9-18. IEEE, 2022.

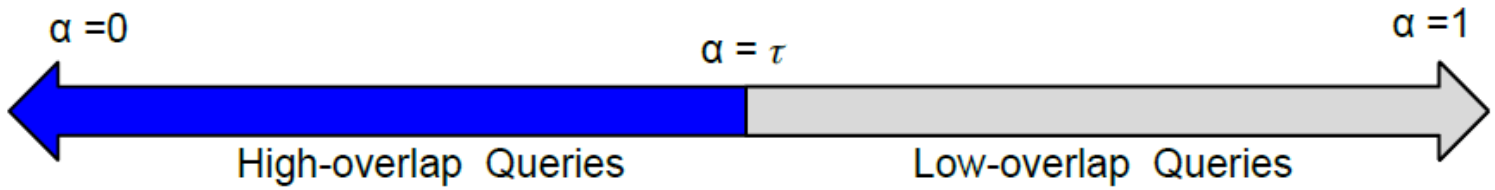
Challenges

- How can we transform and formulate the R-tree search operation into a multi-label classification task?
- As there is no readily available training data, what will be the process of training data preparation for a particular query workload?
- How to process queries efficiently over the ML-enhanced R-tree?
- What is the impact of the choice of ML models on dynamic query workloads?

PURDUE UNIVERSITY®

Definition: Overlap Ratio

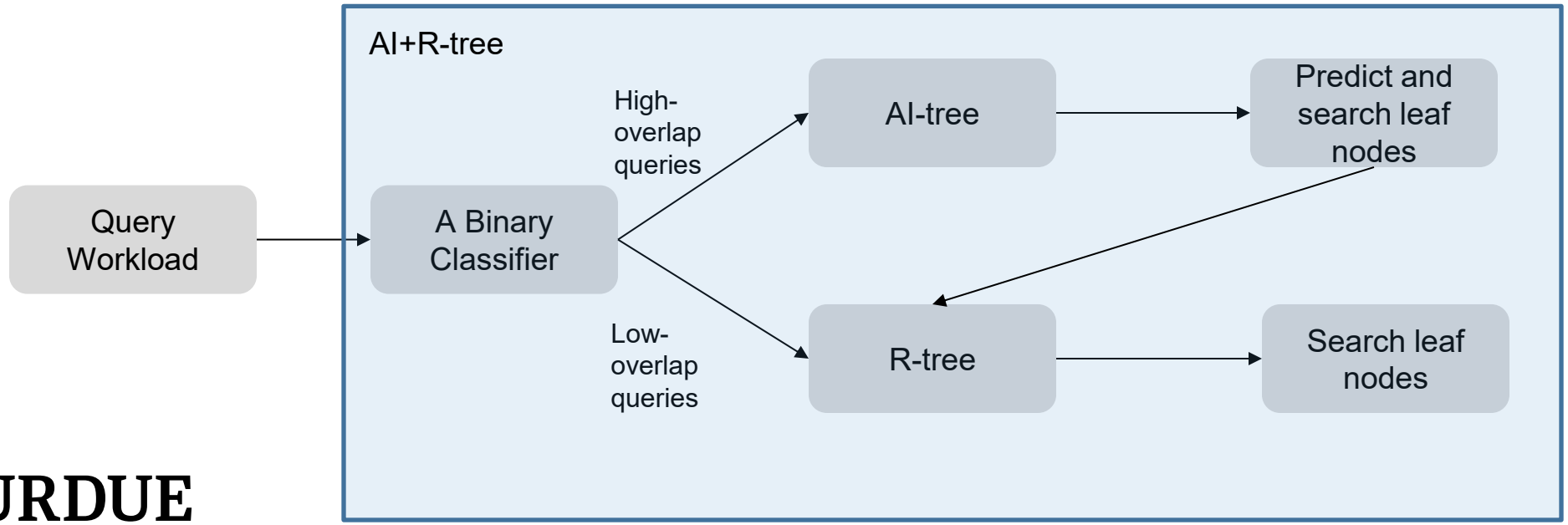
- An overlap ratio α quantifies the degree of extraneous leaf node accesses required by a range query.
- For a range query:
  - $\alpha = \frac{Number\ of\ true\ leaf\ nodes\ that\ actually\ contains\ the\ output\ data\ objects}{Number\ of\ leaf\ nodes\ searched\ by\ the\ R-Tree}$
  - α is in the range [0,1]
- High-overlap Queries:
  - A range query is declared as a "high-overlap query" if the value of α is less than a pre-defined threshold τ.



PURDUE UNIVERSITY

The AI+R-tree

- Query Processing:
  - Leveraging a binary classifier to automatically differentiate between high- and low-overlap queries
  - Process the high-overlap queries using the AI-tree
  - Process the low-overlap queries using the R-tree
  - Adopting a hybrid approach, namely the AI+R-tree, to process both types of range queries efficiently
  - In the worst-case scenario, if the AI-tree returns an empty set, the R-tree is invoked.



PURDUE UNIVERSITY

Limitations of the AI+R-tree

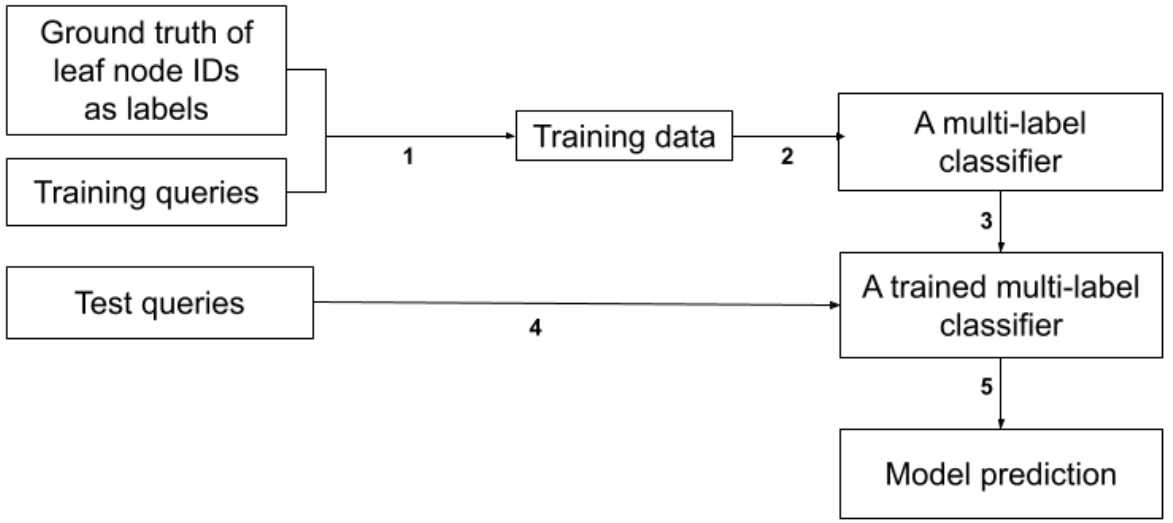
- Operates on static query workloads
- Does not analyze the impact of ML models on query processing performance
- Does not investigate the potential of designing custom ML models for the AI+R-tree

- New contributions:
  - Extend the AI+R-tree to process dynamic query workloads
  - Present the empirical tradeoffs in query processing by varying the choice of ML models
    - Query Recall
    - Query Processing Time
    - ML Model Size
  - Include a case study of designing a custom loss function tailored to the query processing requirements of the AI+R-tree

PURDUE UNIVERSITY

Learning the R-tree Index

- Workflow of ML model training and testing:
  - Prepare training data for a particular data and query workloads
  - Train a multi-label classifier (e.g., multi-label decision tree classifier)
  - Invoke the trained multi-label classifier to directly predict the true leaf node IDs that contain the query result



PURDUE UNIVERSITY

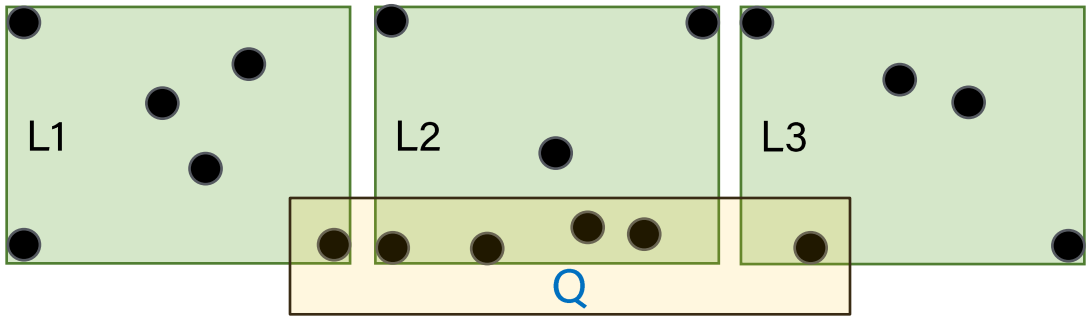
Choice of ML Models for the Multi-label Classifier

- Decision Tree (DT)
- Random Forest (RF)
- eXtreme Gradient Boosting (XGBoost)
- Neural Network (NN)
  - *Variant-1:* NN model with a standard loss function
  - *Variant-2:* NN Model with a custom loss function

PURDUE UNIVERSITY

Designing a Custom Loss Function for the NN-based Models

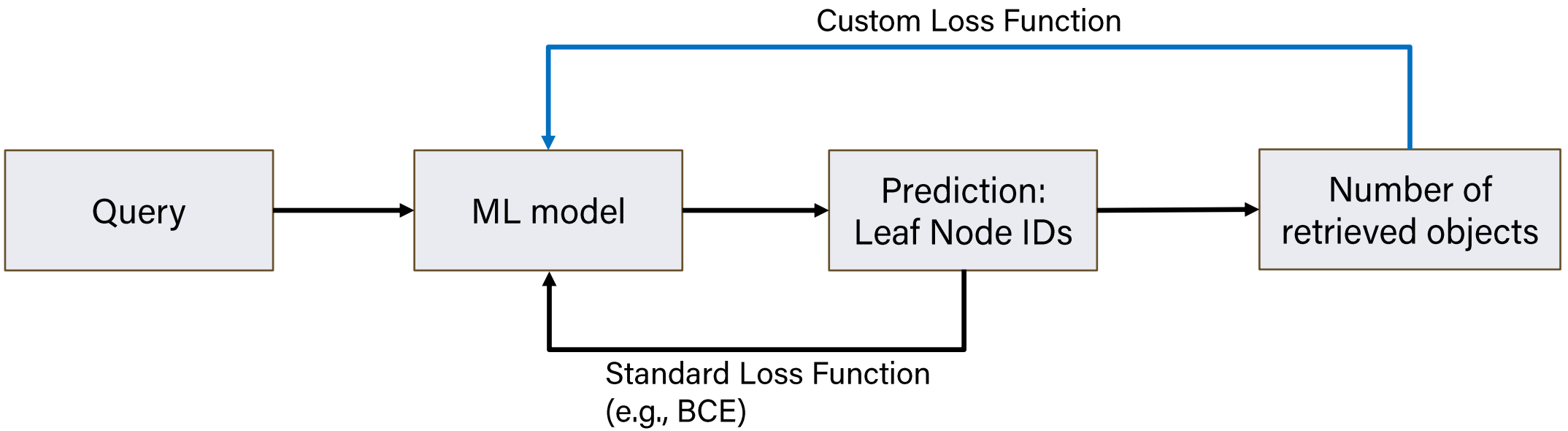
- An input range query: Q
- Ground truth labels (i.e., leaf node ids):
  - $L1$, $L2$, $L3$
- The number of qualifying data objects within $L1$, $L2$, $L3$ (e.g., inside Q) are 1, 4, 1, respectively.
- If an ML model predicts:
  - Only L1 or L3 correctly, the recall is 1/6 = 0.16
  - Only L2 correctly, the recall is 4/6 = 0.66
- *This information is not captured if we calculate the NN model training loss in terms of leaf node predictions only.*



PURDUE UNIVERSITY

Designing a Custom Loss Function for the NN-based Models (Cont'd)

- For the custom loss function, we need to use information beyond the model prediction.



PURDUE UNIVERSITY

Evaluation: Datasets and Query Workloads

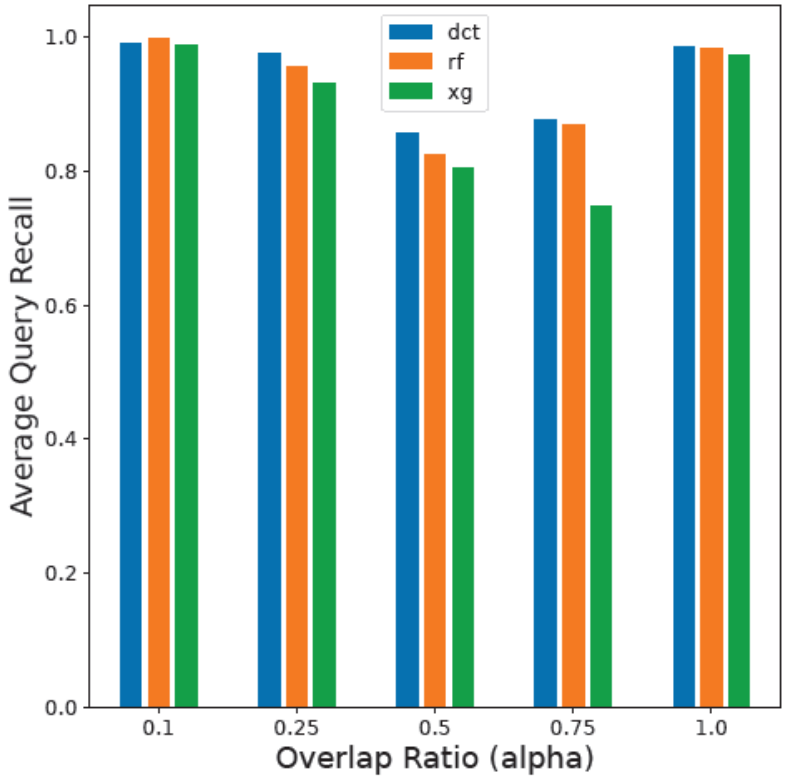
- Three datasets from the UCR Spatio-Temporal Active Repository (UCR-STAR):
  - Tweet locations
  - Gowalla
  - Chicago crimes
- Query workloads:
  - Selectivity: [0.00005, 0.0001, 0.0002]
  - Overlap Raito (α): [0.1, 0.25, 0.5, 0.75, 1.0]
  - Splitting each query set into 60%/20%/20% (train/validation/test)

PURDUE UNIVERSITY®

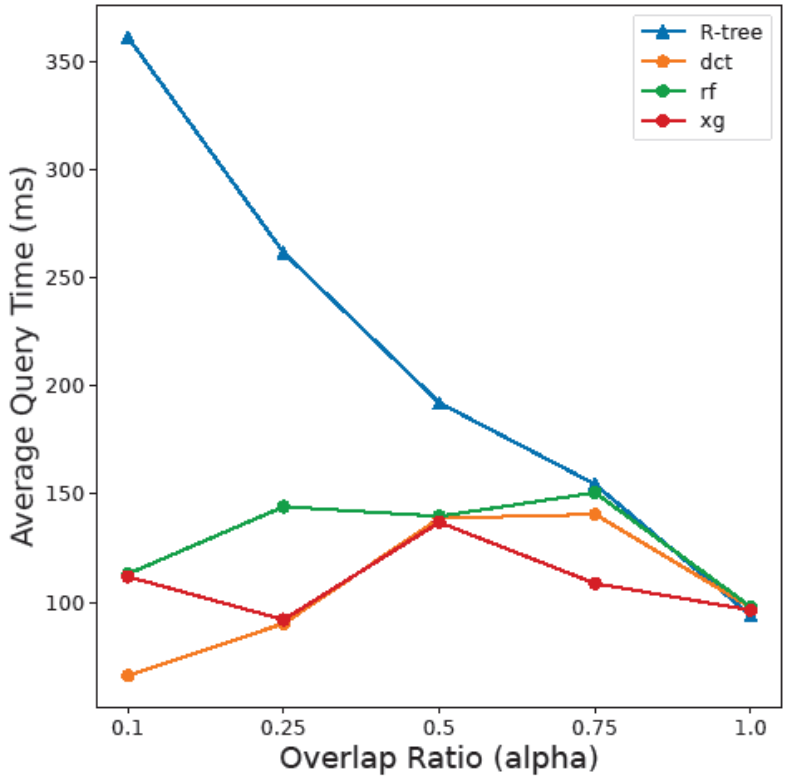
Saheli Ghosh, Tin Vu, Mehrad Amin Eskandari, and Ahmed Eldawy. 2019. UCRSTAR: The UCR spatio-temporal active repository. SIGSPATIAL Special 11, 2 (2019), 34–40.

Evaluation: DT-based Classifiers for the Tweet Locations Dataset

Query Selectivity: 0.0002



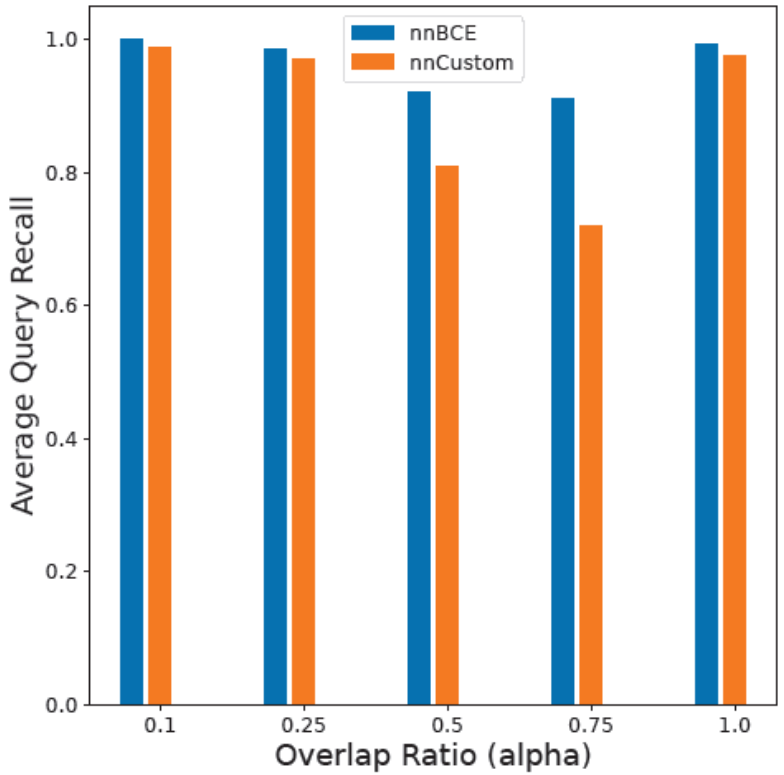
Query Selectivity: 0.0002



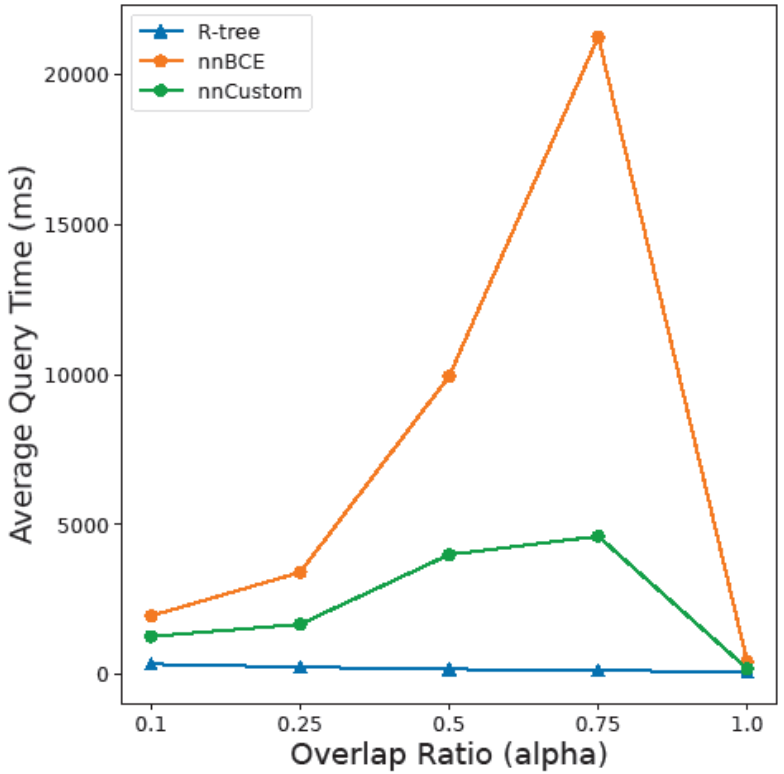
PURDUE UNIVERSITY®

Evaluation: NN-based Classifiers for the Tweet Locations Dataset

Query Selectivity: 0.0002



Query Selectivity: 0.0002



PURDUE UNIVERSITY

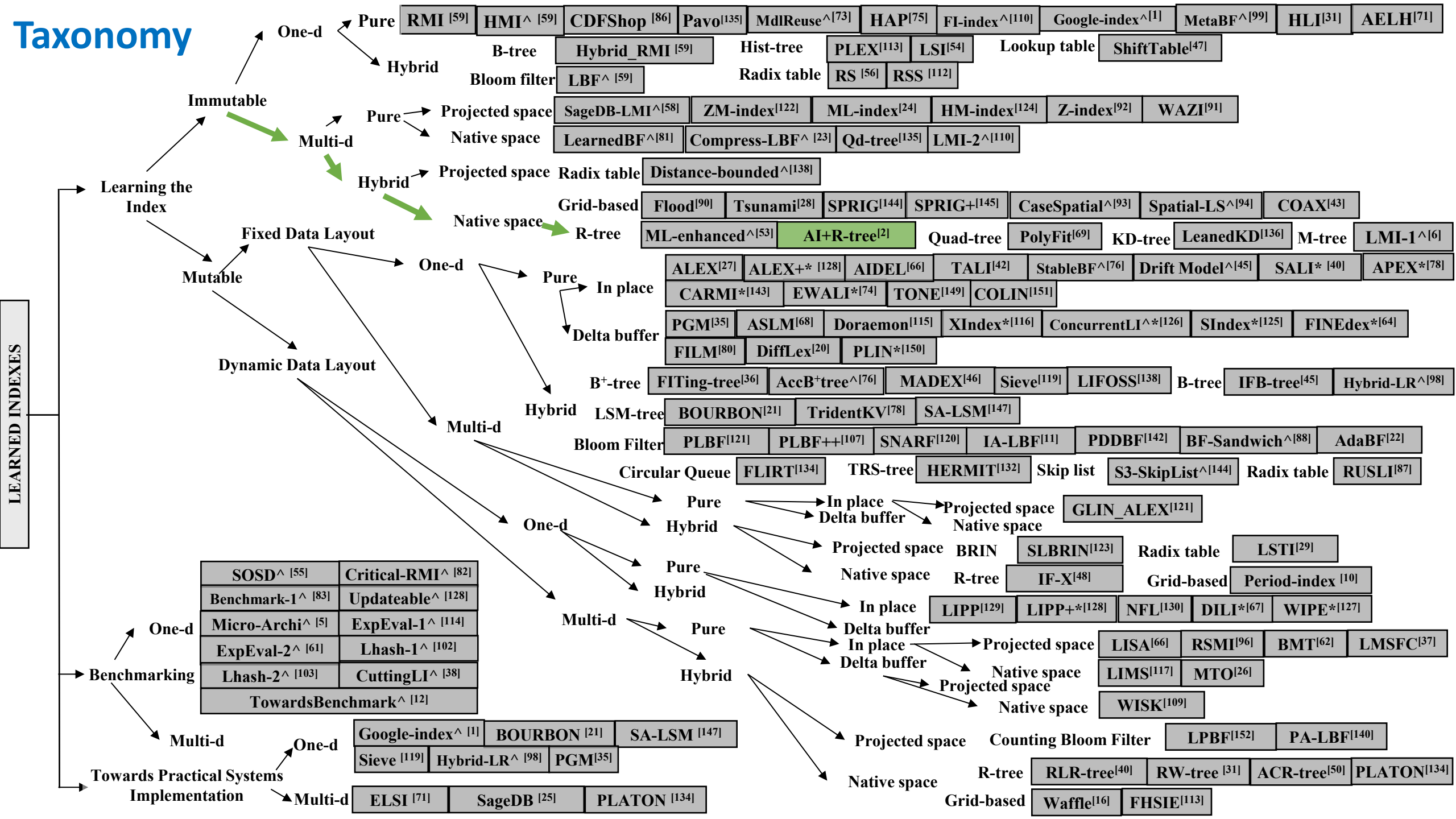
Evaluation: Space Consumption of the ML Models

- Average ML model size of the AI+R-tree for the tweet locations dataset across all α values (in MBs)

		DT-based models			NN-based models	
Selectivity	R-tree	DCT	RF	XG	nnBCE	nnCustom
0.00005	1106.54	2.91	3.44	0.12	0.13	0.13
0.0001	1106.54	2.80	3.44	0.12	0.13	0.13
0.0002	1106.54	2.61	3.44	0.12	0.13	0.13

PURDUE UNIVERSITY

Taxonomy


LEARNED INDEXES
Learning the Index
Immutable
One-d
Pure
RMI [59]
Hybrid
HMI^ [59]
B-tree
Hybrid_RMI [59]
Bloom filter
LBF^ [59]
Multi-d
Pure
Projected space
SageDB-LMI^[58]
ZM-index[122]
ML-index[24]
HM-index[124]
Z-index[92]
WAZI[91]
Native space
LearnedBF^[81]
Compress-LBF^ [23]
Qd-tree[135]
LMI-2^[110]
Hybrid
Projected space
Radix table
Distance-bounded^[138]
Native space
Grid-based
Flood[90]
Tsunami[28]
SPRIG[144]
SPRIG+[145]
CaseSpatial^[93]
Spatial-LS^[94]
COAX[43]
R-tree
ML-enhanced^[53]
AI+R-tree[2]
Mutable
Fixed Data Layout
One-d
Pure
In place
ALEX[27]
ALEX+* [128]
AIDEL[66]
TALI[42]
StableBF^[76]
Drift Model^[45]
SALI* [40]
APEX*[78]
CARMI*[143]
EWALI*[74]
TONE[149]
COLIN[151]
Delta buffer
PGM[35]
ASLM[68]
Doraemon[115]
XIndex*[116]
ConcurrentLI^*[126]
SIndex*[125]
FINEdex*[64]
FILM[80]
DiffLex[20]
PLIN*[150]
Hybrid
B+-tree
FITing-tree[36]
AccB+tree^[76]
MADEX[46]
Sieve[119]
LIFOSS[138]
B-tree
IFB-tree[45]
Hybrid-LR^[98]
LSM-tree
BOURBON[21]
TridentKV[78]
SA-LSM[147]
Multi-d
Bloom Filter
PLBF[121]
PLBF++[107]
SNARF[120]
IA-LBF[11]
PDDBF[142]
BF-Sandwich^[88]
AdaBF[22]
Circular Queue
FLIRT[134]
TRS-tree
HERMIT[132]
Skip list
S3-SkipList^[144]
Radix table
RUSLI[87]
Dynamic Data Layout
One-d
Pure
In place
Projected space
GLIN_ALEX[121]
Delta buffer
Native space
Hybrid
Projected space
BRIN
SLBRIN[123]
Radix table
LSTI[29]
Native space
R-tree
IF-X[48]
Grid-based
Period-index [10]
Pure
In place
LIPP[129]
LIPP+*[128]
NFL[130]
DILI*[67]
WIPE*[127]
Delta buffer
Hybrid
Multi-d
Pure
In place
Projected space
LISA[66]
RSMI[96]
BMT[62]
LMSFC[37]
Delta buffer
Native space
LIMS[117]
MTO[26]
Hybrid
Projected space
Native space
WISK[109]
Projected space
Counting Bloom Filter
LPBF[152]
PA-LBF[140]
Native space
R-tree
RLR-tree[40]
RW-tree [31]
ACR-tree[50]
PLATON[134]
Grid-based
Waffle[16]
FHSIE[113]
Benchmarking
One-d
SOSD^ [55]
Critical-RMI^ [82]
Benchmark-1^ [83]
Updateable^ [128]
Micro-Archi^ [5]
ExpEval-1^ [114]
ExpEval-2^ [61]
Lhash-1^ [102]
Lhash-2^ [103]
CuttingLI^ [38]
TowardsBenchmark^ [12]
Multi-d
Towards Practical Systems Implementation
One-d
Google-index^ [1]
BOURBON [21]
SA-LSM [147]
Sieve [119]
Hybrid-LR^ [98]
PGM[35]
Multi-d
ELSI [71]
SageDB [25]
PLATON [134]

Conclusion and Future Research Directions

- The AI+R-tree combines both the traditional R-tree structure and the learned R-tree (i.e., the AI-tree) structure to maximize query processing performance.

- Overall, this study takes an important step towards realizing the AI+R-tree in various practical settings, and opens several directions for future research:
  - Investigating the benefit of the AI+X-tree structure, where X is a traditional spatial or a multi-dimensional index structure with the issue of node overlapping.
  - Designing a mutable AI+R-tree, and developing efficient query processing techniques over the mutable structure
  - Optimizing the proposed custom NN-based classifier for improving the query processing latency

PURDUE UNIVERSITY®

THANK YOU

PURDUE UNIVERSITY®