Adversarial Workload Matters

Executing a Large-Scale Poisoning Attack against Learned Index Structures

Matthias Bachfischer (Student ID: 1133751)

Melbourne, October 18th 2021

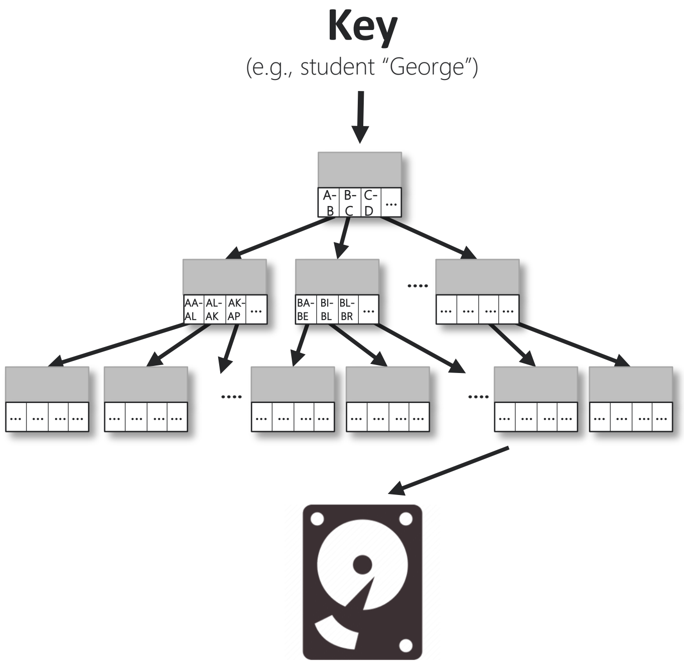
Agenda

1. **Introduction**
2. **Background on Adversarial ML**
3. **Poisoning Attacks on Learned Index Structures**
4. **Experimental Setup**
5. **Experimental Results**
6. **Conclusion**

Agenda

Traditional DB indexes use tree data structures to find record on disk – learned indexes uses ML models to "predict" location

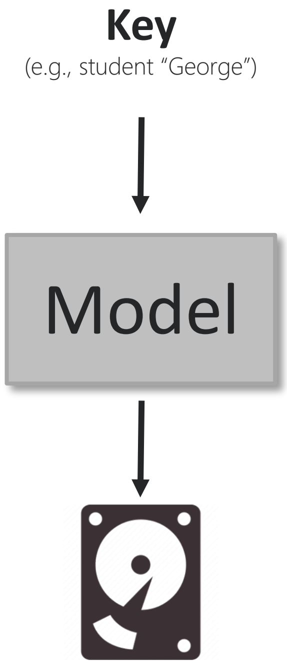
Traditional database indexes



Type: B-Tree (and variantes)

Complexity: $O(log_2 n)$

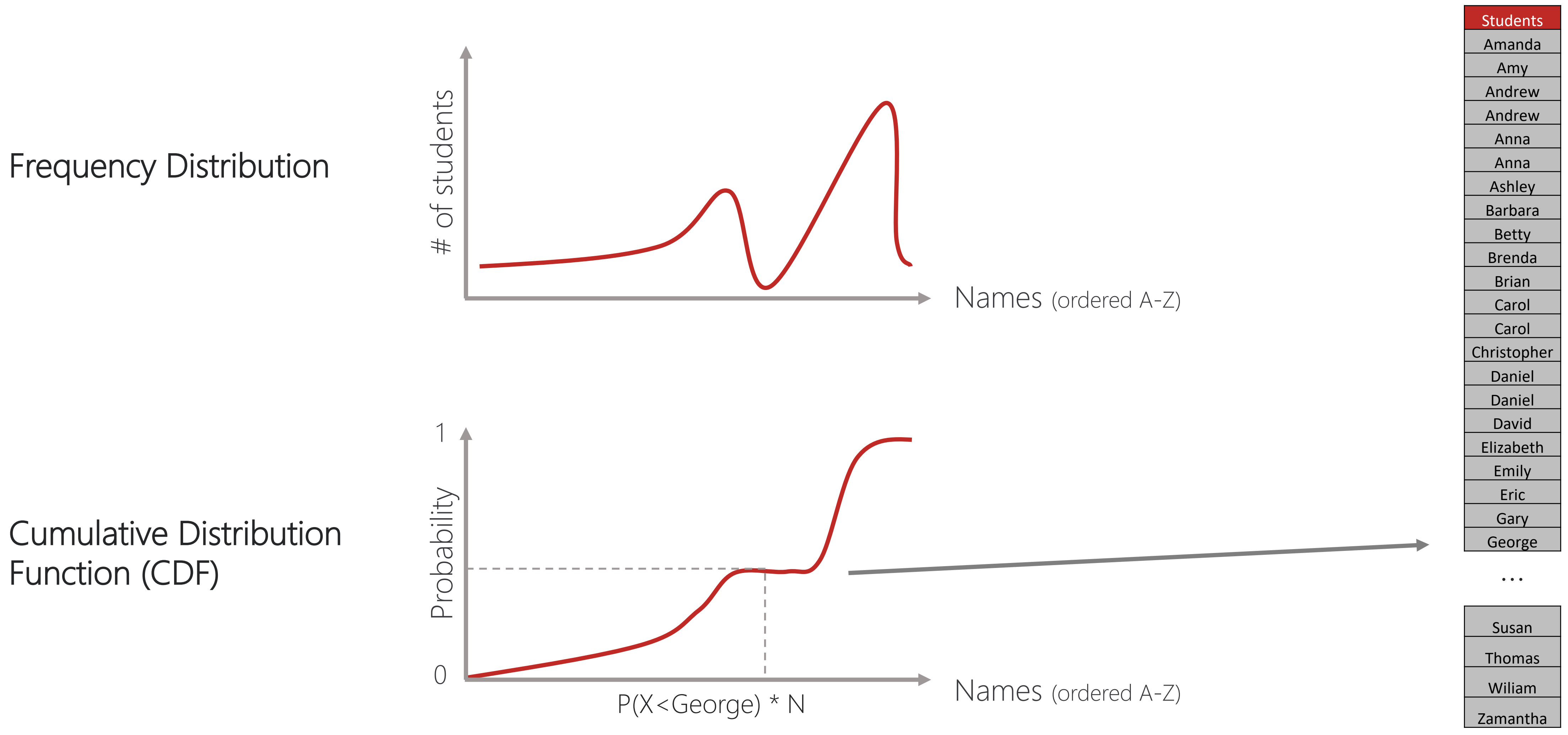
Learned Index Structures (LIS)



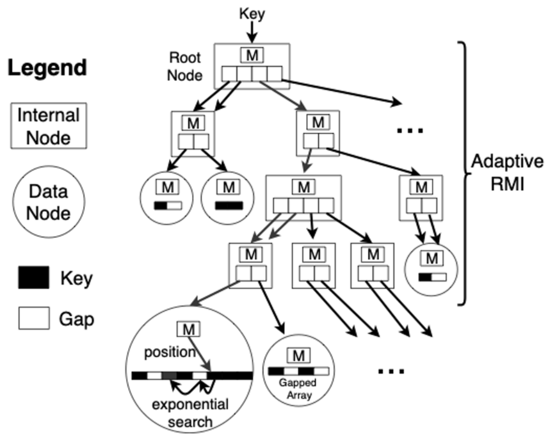
Type: Linear regression (and variantes)

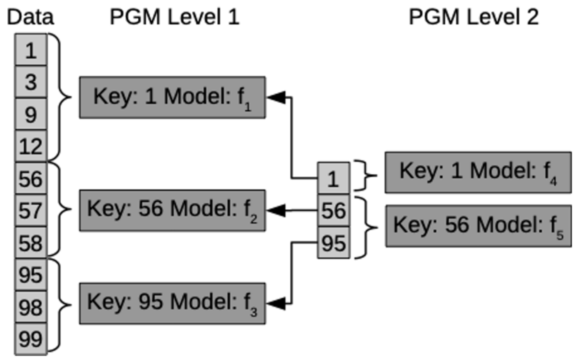
Complexity: $O(1)$

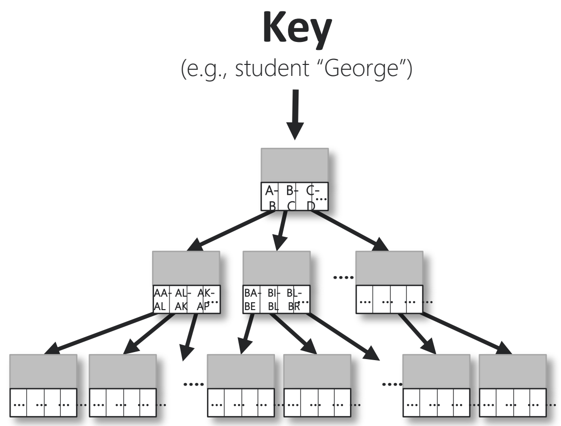
A learned index structure works by approximating the CDF of the data to predict the location of the query key



This research has evaluated three index implementations: Two learned index structures (ALEX & D-PGM) and one B+Tree







	ALEX	Dynamic-PGM	B+Tree
Paper:	ALEX: an updatable adaptive learned index	The PGM-index: a fully-dynamic compressed learned index with provable worst-case bounds	STX B+ Tree C++ Template Classes v0.9
Authors:	Ding et al. (2019)	Ferragina et al. (2020)	Timo Bingmann (2007)
Model:	Piecewise Linear Approximation (PLA)	Piecewise Linear Approximation (PLA)	None
Code:	https://github.com/microsoft/ALEX	https://github.com/gvinciguerra/PGM-index	https://github.com/bingmann/stx-btree

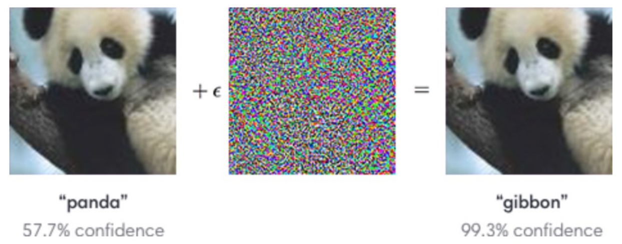
Agenda

Adversarial Machine Learning is the study of ML techniques against an adversarial opponents aiming to fool the model

Attacker goals

Attacker capability

	Integrity	**Availability**	**Privacy / Confidentiality**
	Misclassifications that do not compromise normal system operation		...ies that reveal secret ...e model or its users
Test data	Evasion (a.k.a. adversarial examples)		...action / stealing ...del inversion ...climbing attacks)
Training data	Poisoning to allow subsequent intrusions (a.k.a. backdoors or neural network trojans)	Focus of this work: Poisoning to maximize classification error (a.k.a. worst-case behavior)	n/a



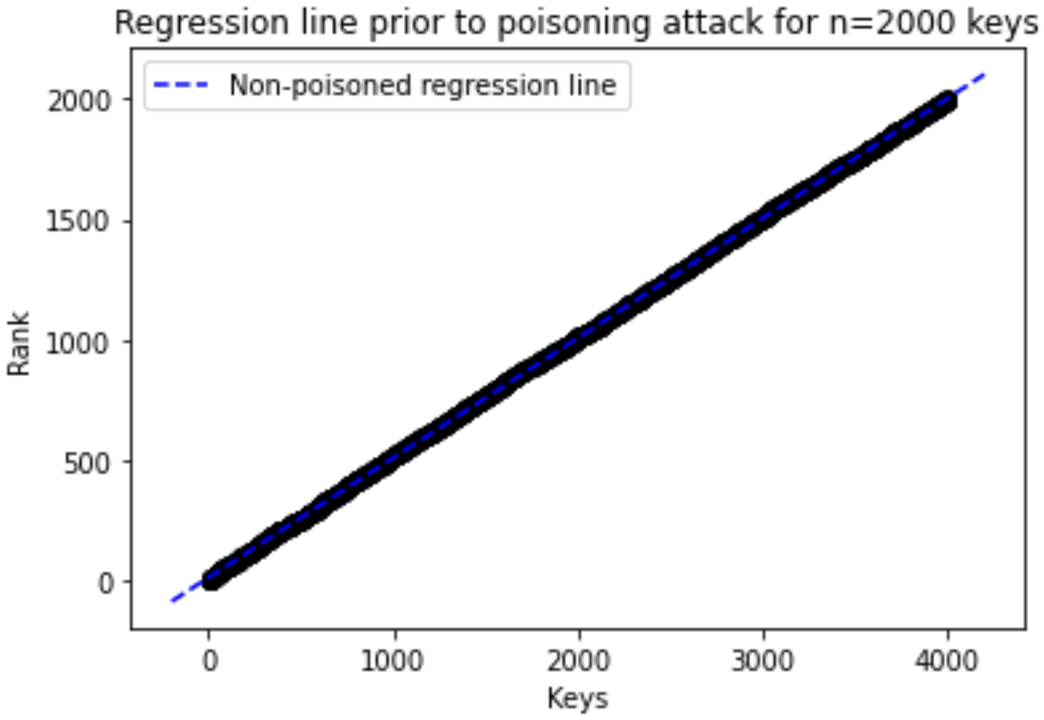
An adversarial input, overlaid on a typical image, can cause a classifier to miscategorize a panda as a gibbon.

Source: Biggio, B. and F. Roli (2018). *Wild patterns: Ten years after the rise of adversarial machine learning*. Pattern Recognition 84: 317-331.

Agenda

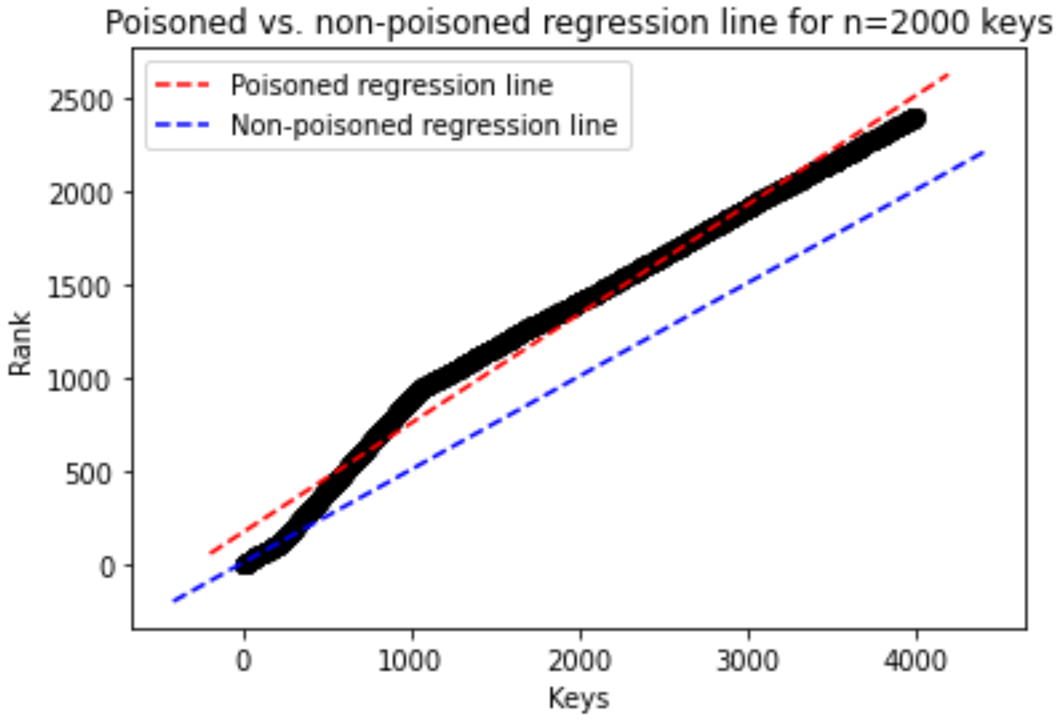
To study the performance of learned indexes under adversarial workload, we execute a poisoning attack against the CDF

1. Dataset **before** poisoning attack:



Mean Squared Error (MSE): 22.99

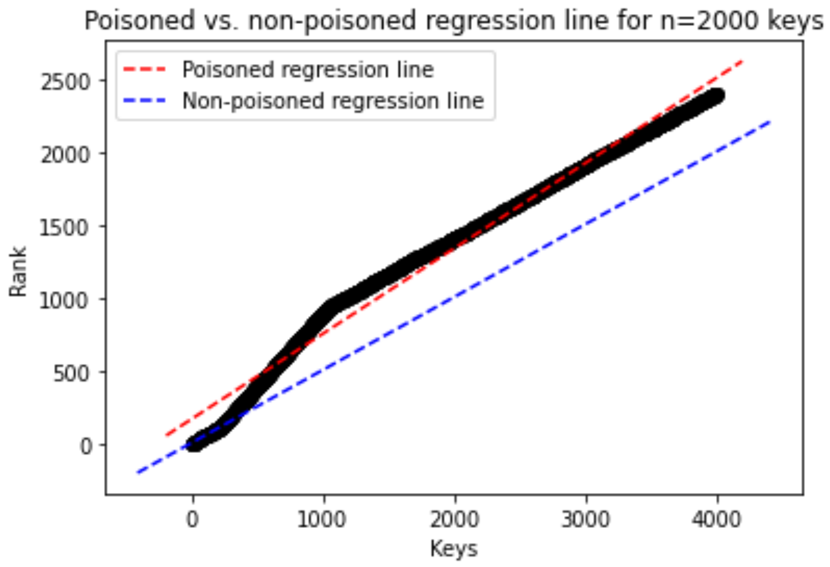
2. Dataset **after** poisoning attack:



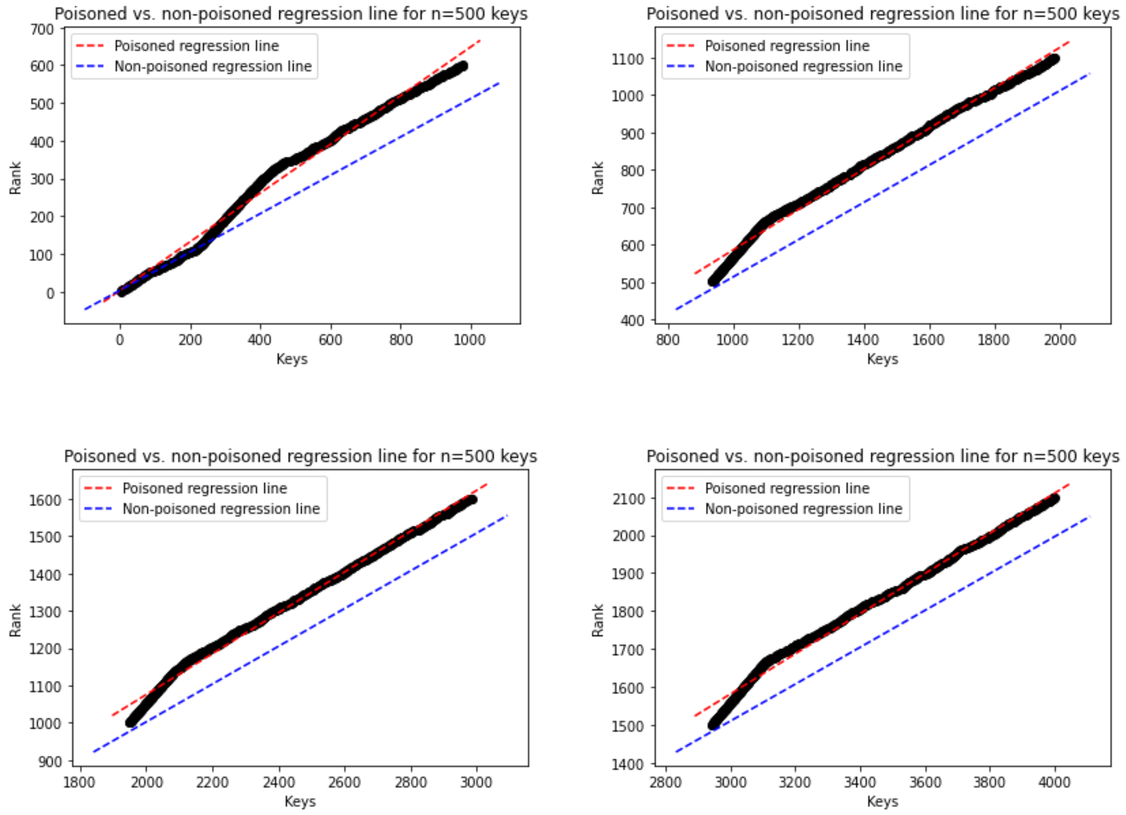
Mean Squared Error (MSE): 8675.83

Poisoning threshold p = 20% (400 keys)

Distributed poisoning algorithm deliberately places poisoning keys in dense areas to exacerbate the non-linearity of the CDF



Algorithm: GreedyPoisoningRegressionCDF[1]
(based on Kornaropoulos et al.)

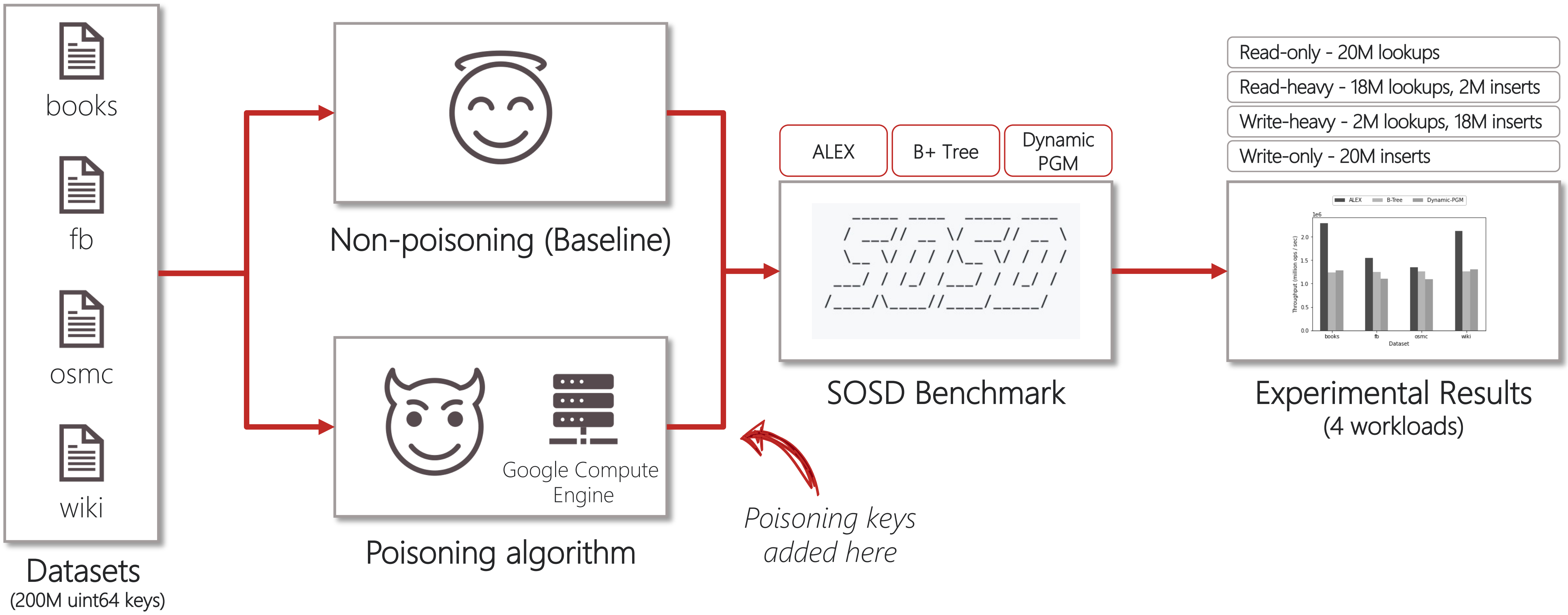


Algorithm: Greedy**Distributed**PoisoningRegressionCDF

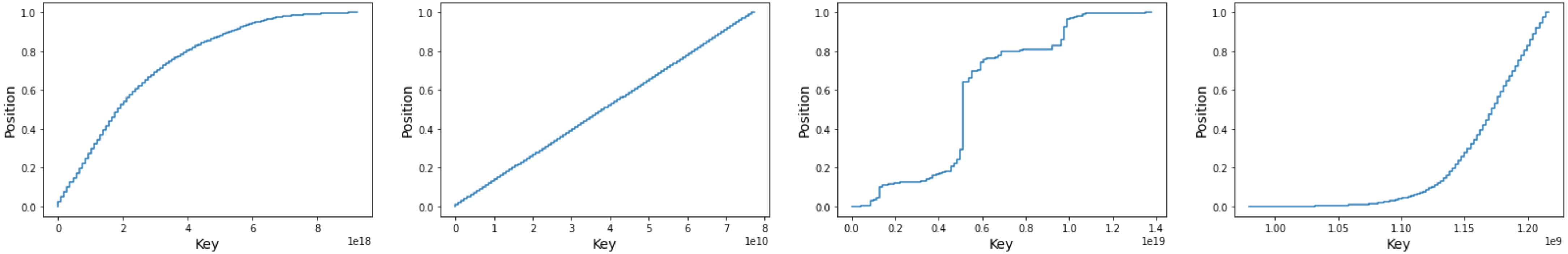
1) Kornaropoulos, E. M., et al. (2020). *The Price of Tailoring the Index to Your Data: Poisoning Attacks on Learned Index Structures.*

Agenda

Data pipeline extends existing open-source tools from learned indexes research community

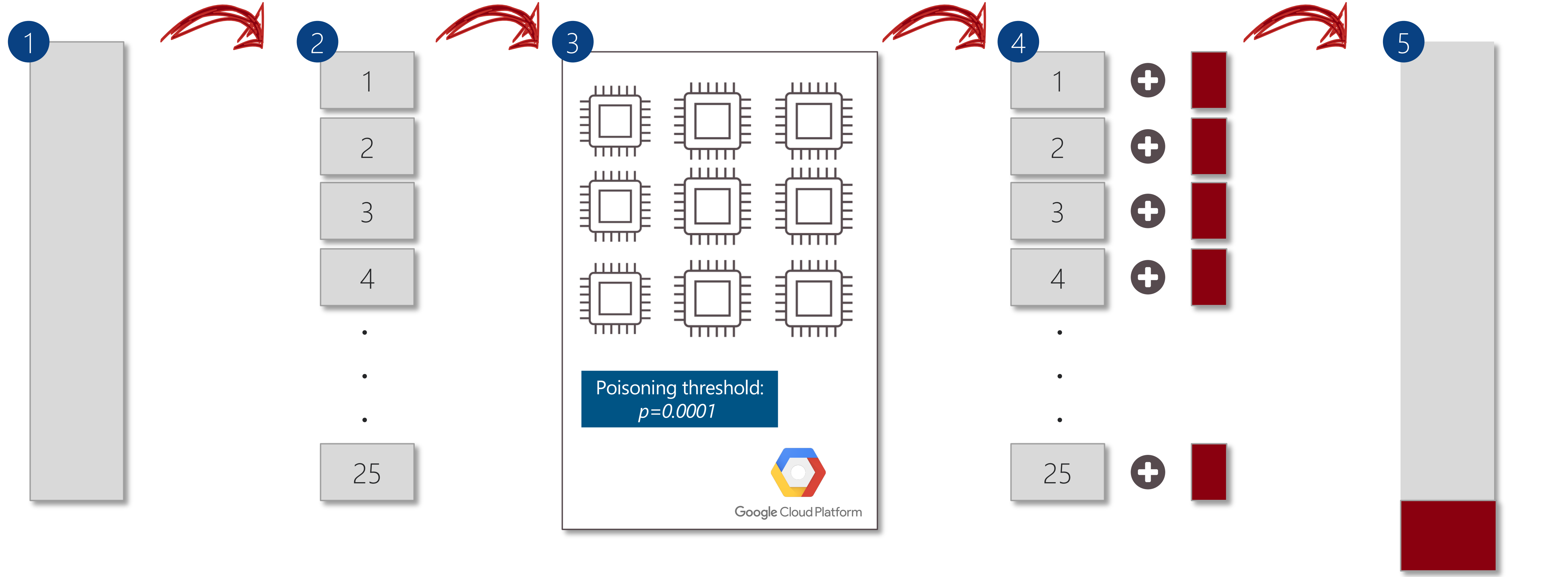


Key distribution of SOSD benchmark datasets varies heavily in terms of their cumulative distribution function



	books_200M_uint64	fb_200M_uint64	osm_cellids_200M_uint64	wiki_ts_200M_uint64
Contents:	Book popularity on Amazon	Facebook user IDs	Cell IDs on Open Street Map	Wikipedia edit timestamps
Data:	200M uint64 keys	200M uint64 keys	200M uint64 keys	200M uint64 keys
Size:	1.53 GB	1.53 GB	1.53 GB	1.53 GB

Poisoning attack was scaled to target up to 200M keys by exploiting the power of a very large virtual machine



200M legitimate keys (full dataset)

25x 80M legitimate keys (data chunks)

VM instance c2-standard-30 (30 vCPUs, 120 GB RAM)

25x 80M legitimate keys + 25 x 800 poisoning keys

200M legitimate keys + 20k poisoning keys

Note: Poisoning attack was executed on a *c2-standard-30* VM instance running on Google Compute Cloud (30 vCPUs, 120GB RAM).

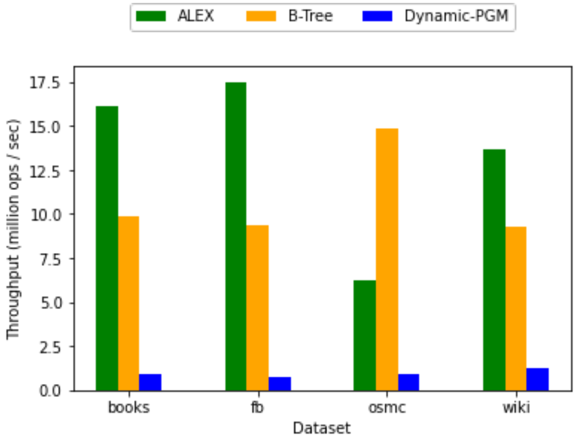
Agenda

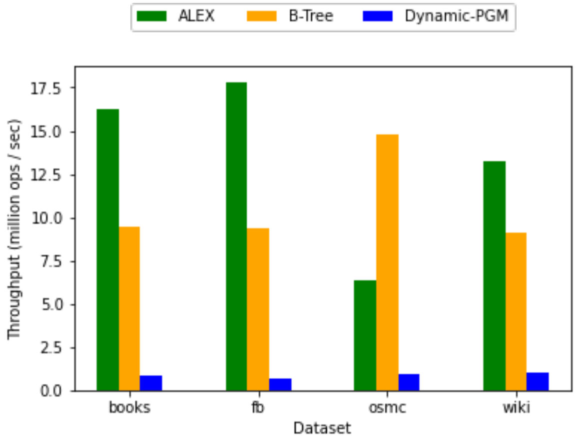
Final results from poisoned and non-poisoned workload scenarios with poisoning percentage p=0.0001 (I/II)

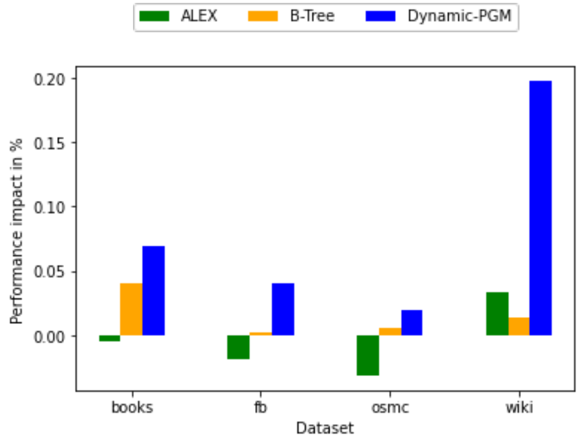
Non-poisoned scenario: Poisoned scenario: Performance impact:

1.
Read-only workload

20M lookups

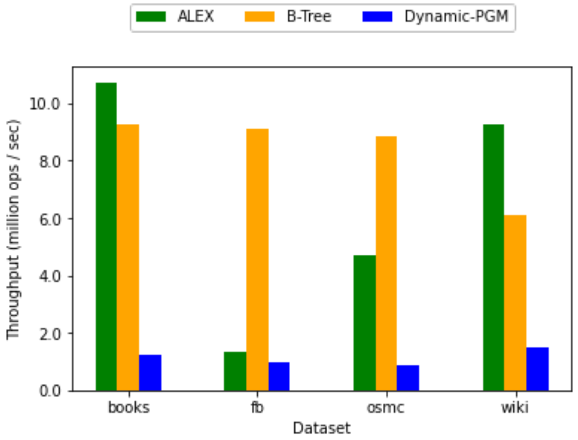


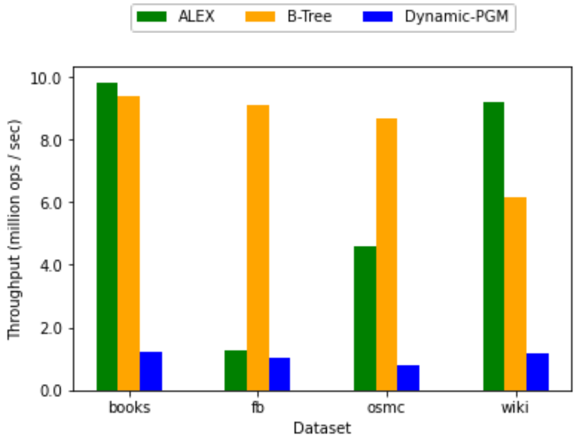


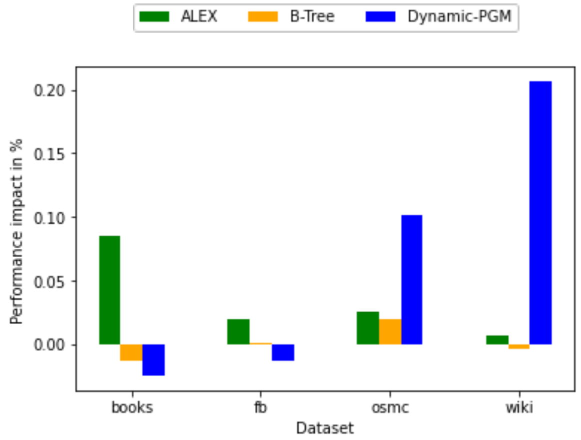


2.
Read-heavy workload

18M lookups, 2M inserts







Note: All datasets in this experiment consist of 200M uint64 keys. The benchmark was evaluated on a *e2-standard-8* VM instance running on Google Compute Cloud (8 vCPUs, 32GB RAM). For each index, multiple configurations were tested..

Final results from poisoned and non-poisoned workload scenarios with poisoning percentage p=0.0001 (II/II)

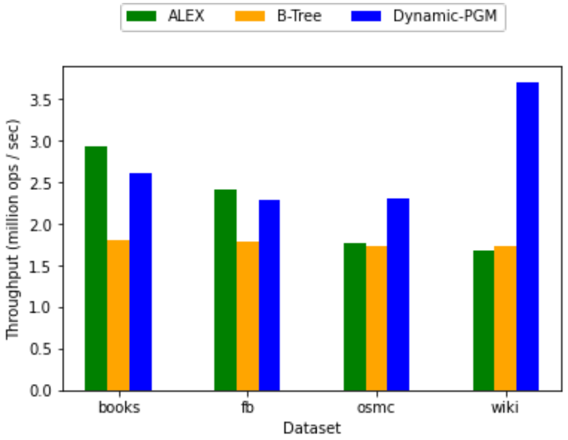
Non-poisoned scenario:

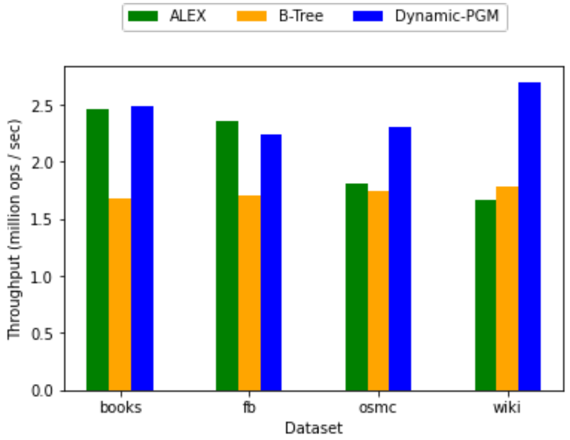
Poisoned scenario:

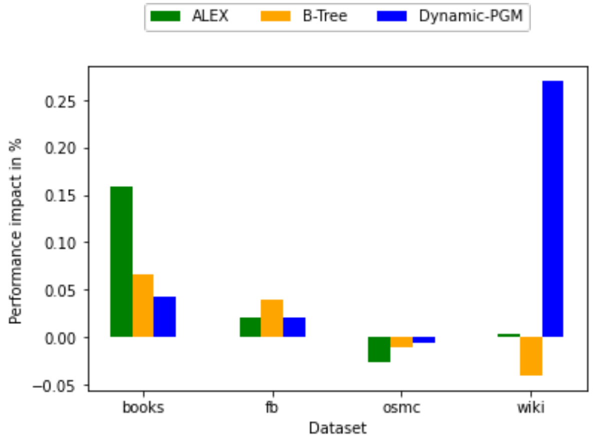
Performance impact:

3.
Write-heavy workload

2M lookups, 18M inserts

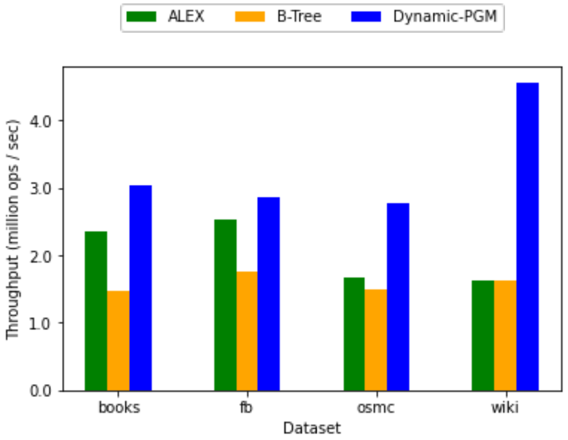


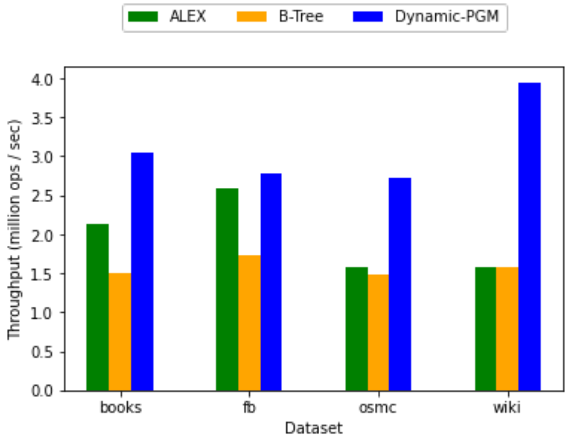


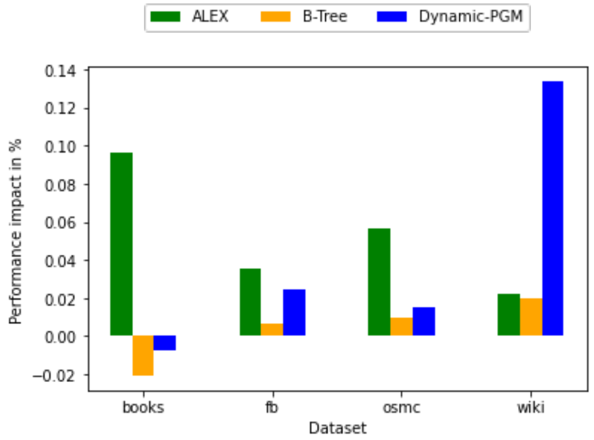


4.
Write-only workload

20M inserts







Note: All datasets in this experiment consist of 200M uint64 keys. The benchmark was evaluated on a e2-standard-8 VM instance running on Google Compute Cloud (8 vCPUs, 32GB RAM). For each index, multiple configurations were tested.

Agenda

Research has shown that “adversarial workload matters”: Up to 20% performance deterioration observed in learned indexes

1

Limitations

- Poisoning attack executed with poisoning threshold of $p = 0.0001$ due to computational constraints (vs. $p = 0.01$ to $p = 0.2$)
- Experiments performed on Virtual Machine instance (*e2-standard-8*), thus sensitive to I/O demand on physical host

2

Recommendations

- Various approaches for defending against poisoning attacks on linear regression (TRIM[1], LID[2]), but not directly applicable
- In LIS, the rank for each key depends on value of all other keys (mitigations needs to iteratively re-calibrate!)
- Removal of poisoning keys often difficult (concentrated around legitimate keys)

3

Future Directions

- Assess other models for constructing learned index structures, such as polynomial interpolation[3] or logarithmic error regression[4]
- Leverage the update functionality of LIS to insert / delete keys at runtime

1) Jagielski, M., et al. (2018). *Manipulating machine learning: Poisoning attacks and countermeasures for regression learning.* 2018 IEEE Symposium on Security and Privacy (SP), IEEE
2) Weerasinghe, S., et al. (2020). *Defending regression learners against poisoning attacks.* arXiv preprint arXiv:2008.09279.
3) Setiawan, N. F., et al. (2020). *Function interpolation for learned index structures.* Australasian Database Conference, Springer.
4) Eppert, M., et al. (2021). *A Tailored Regression for Learned Indexes: Logarithmic Error Regression.* Fourth Workshop in Exploiting AI Techniques for Data Management.