2019/9/23

# Questions for Lecture 3 - Indexing

Daren Chao

---

## The Case for Learned Index Structures

1. B-trees map a key to a position with a min- and max- error. In ML model, we need to calculate and remember the worst over- and under- prediction of a position to estimate error. What if the range is too large?
Even if few values have a big error, it will affect the result greatly. (Section 2)

2. This work is designed for read-only databases.
Is there any inspiration for updatable databases or even insert-heavy workload situation?

3. The new system, Learned Index Framework (LIF), never use Tensorflow at inference; rather, it extracts all weights from the trained TF model and guarantee efficient index structures. All unnecessary overhead and instrumentation are removed.
How does it work? (Section 3)

4. This article mentioned that only the monotonic model can use the range index. Why? (Section 2)

5. How is the learned hash-map trained? (Section 4)

---

## SageDB: A Learned Database System

1. The paper mentioned the semantic guarantees several times.
What exactly does it mean in this paper?
Why traditional ML can not meet it but this work can?
How can they accomplish it?

2. For multi-dimension data, projections are required.
How does this paper choose proper projection approaches? (Section 3.2)

---

## Considerations for Handling Updates in Learned Index Structures

1. In this paper, when deleted data becomes too much, how can they deal with these flagged data? Should it be reused? (Section 2)

2. When too many nodes are inserted, will the interpolation of reference point be inaccurate? (Section 3)

3. The efficiency evaluation is not given in detail in the paper.
What situation does this approach apply? In what cases is it better than B-tree, and is there any case that it even worse than traditional methods?

---

## A Model for Learned Bloom Filters and Related Structures

1. Traditional Bloom Filters guarantee a FNR of 0. How does this paper solve this problem?

2. Will the Learned Bloom filter has a higher time complexity?

---

## ALEX: An Updatable Adaptive Learned Index

1. When too many nodes are inserted, will the "tree" of model become deep?
How to prevent this layer from being too deep and how can they deal with it? (Section 3)

2. What is the meaning of the output of each stage?
Will each stage narrow down the range **or** only used to specify which model to use for the next stage?
If the latter, why not try using more information, i.e., passing information to next layers?

---

Acknowledgement:
Some of these questions were proposed during the discussions with Yibin Zhang.