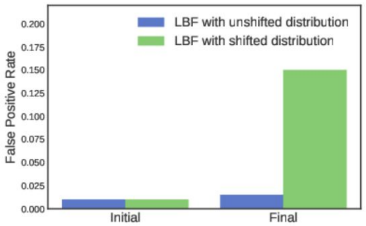
Adjusting Bloom Filters for Distribution Shift with Semantic Hashing

Problem Statement:

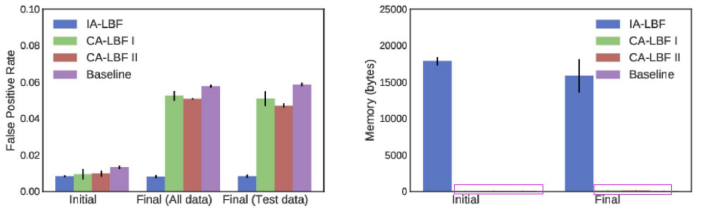
The competitive advantage of learned bloom filters (Kraska, Tim, et al.) and Ada-BF (Dai, Z., & Shrivastava, A) relies entirely on the model understanding a fixed target distribution. This is a general assumption across much of machine learning, but fails drastically in this particular domain. In common BF cases, such as caching, username checking, and general data storage, distributions may rapidly change when inserting new keys, degrading model performance. We need new techniques that are more amenable to both key and query distribution changes.



Experiment from Bhattacharya et al.

Existing Solutions:

Existing solutions to key-query distribution shift involve adding more filters or retraining the classifier model (Bhattacharya et al.) and still struggle to improve performance (baseline of an LBF) without drastic memory increases.



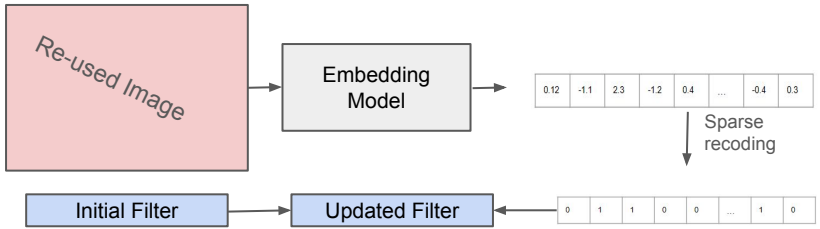
(a) False positive rate on entire data **(b) Memory usage**

Experiment from Bhattacharya et al. Pink boxes (our emphasis) showcase memory baselines exceeded for improvement.

Our Solution:

We propose a Bloom Filter that utilizes *semantic hashing*, where semantically similar queries are hashed to nearby regions in the bloom filter. This reproduces the effect of a prediction model while also robustly handling distribution shift and allowing arbitrary key insertion.

We transform an embedding into a sparse 0-1 vector suitable for hashing, facilitating efficient storage without needing substantial memory increases.



Anticipated Challenges:

We aim to provide **practical** performance results on large image datasets and text datasets and **theoretical** guarantees assuming a suitable embedding distribution.

We additionally anticipate practical challenges in effectively utilizing the entire bloom filter in cases where the distribution lies in a limited region of the embedding space.

Devan Shah and David Yan