P&S Modern SSDs
DeepSketch:
A New Machine Learning-Based
Reference Search Technique
for Post-Deduplication Delta Compression

Jisung Park\textsuperscript{1*}, Jeonggyun Kim\textsuperscript{2*}, Yeseong Kim\textsuperscript{2}, Sungjin Lee\textsuperscript{2}, and Onur Mutlu\textsuperscript{1}

\textsuperscript{1} SAFARI
ETH zürich

\textsuperscript{2} DGVIST
Deagou Gyeongbuk Institute of Science & Technology

USENIX FAST 2022

*J. Park and J. Kim are co-primary authors.
Executive Summary

**Motivation**
- **Data reduction:** Effective at reducing the management cost of a data center by reducing the amount of data physically written to storage devices.
- **Post-deduplication delta compression:** Maximizes the data-reduction ratio by applying delta compression along with deduplication and lossless compression.

**Problem:** Existing post-deduplication delta-compression techniques provide significantly low data-reduction ratios compared to the optimal.
- Due to the limited accuracy of reference search for delta compression.
- Cannot identify a good reference block for many incoming data blocks.

**Key Idea:** DeepSketch, a new machine learning-based reference search technique that uses the learning-to-hash method.
- Generates a given data block’s signature (sketch) using a deep neural network.
- The higher the delta-compression benefit of two data blocks, the more similar the signatures of the two blocks to each other.

**Evaluation Results:** DeepSketch reduces the amount of physically-written data.
- Up to 33% (21% on average) compared to a state-of-the-art baseline.
Talk Outline

- Data Reduction in Storage Systems
- DeepSketch: A New Machine Learning-based Reference Search Technique
- Evaluation Results
Big Data Era

- Unprecedented amounts of data processed in modern computing systems
  - e.g., Facebook generates 4 petabytes of new data every day
Data Reduction in Storage Systems

- Effective at reducing the management cost of a data center
Data Reduction in Storage Systems

- Effective at reducing the management cost of a data center
  - By reducing the amount of written data to storage devices
Data Reduction in Storage Systems

- Effective at reducing the management cost of a data center
  - By reducing the amount of written data to storage devices
  - Enabling the system to deal with the same amount of data with fewer and/or smaller storage devices
Post-deduplication Delta Compression

- Combines three different data-reduction approaches
  - To maximize the data-reduction ratio \( (= \frac{\text{Original Data Size}}{\text{Reduced Data Size}}) \)
  - Deduplication → Delta compression → Lossless compression
  - Can achieve more than 2x data reduction over a simple combination of deduplication and lossless compression
Overview of Post-Deduplication Delta Compression

File System

Data Reduction Module

Storage Device
Step 1: Deduplication

File System

Data Reduction Module

Storage Device
Step 1: Deduplication

File System

Data Reduction Module

Storage Device

Deduplication
Step 2: Delta Compression

File System

Data Reduction Module

Storage Device
Step 2: Delta Compression

File System

Data Reduction Module

Storage Device

Reference

\(Y: \text{Delta-compressed}\)
Step 3: Lossless Compression

File System

Data Reduction Module

Storage Device

Y: *Delta-compressed*
Key Challenge: Reference Search

- How to find a good reference block for an incoming data block across a wide range of stored data at low cost

- Scanning all stored data blocks: Prohibitive performance overhead

- Reference search in deduplication
  - Uses a strong hash function (e.g., SHA1 or MD5) to generate a data block’s fingerprint
  - Enables quick reference search by comparing only fingerprints

- Reference search in delta compression
  - Difficult to use a strong hash function that generates significantly different hash values for non-identical yet similar data blocks
State-of-the-Art: Data Sketching

- Generates a data signature (called *sketch*) of each data block
  - **Sketch**: More *approximate* signature than fingerprint
  - **Goal**: two similar data blocks have similar sketches

| Block 1 | U  | S  | E  | N  | I  | X  | F  | A  | S  | T  | 2  | 0  | 2  | 2
|---------|----|----|----|----|----|----|----|----|----|----|----|----|----|---
|         | U  | S  | E  | N  | I  | X  | F  | A  | S  | T  | 2  | 0  | 2  | 2

| Block 2 | U  | S  | E  | N  | I  | X  | F  | A  | S  | T  | 2  | 0  | 2  | 0
|---------|----|----|----|----|----|----|----|----|----|----|----|----|----|---
|         | U  | S  | E  | N  | I  | X  | F  | A  | S  | T  | 2  | 0  | 2  | 0

\[
\begin{align*}
  \text{Feature}_1 &= H_1(\text{SENI}) = 0\times73 \\
  \text{Feature}_2 &= H_2(\text{FAST}) = 0\times32 \\
  \text{Feature}_3 &= H_3(\text{USEN}) = 0\timesF1 \\
  \text{Feature}_4 &= H_4(\text{S202}) = 0\timesCC \\
  \text{Super Feature} &= SF(\text{Block 1}) = 0\times7332F1CC \\
  \text{Super Feature} &= SF(\text{Block 2}) = 0\times7332F1CC
\end{align*}
\]
Limitations of Existing Techniques

- Provide **significantly lower data-reduction ratios** than the optimal
  - Due to **limited accuracy** in reference search for delta compression

- In a general-PC-usage workload, an SF-based approach
  - Provides only **60% of the data-reduction ratio** of brute-force search
  - **High false-negative ratio**: Fails to find any reference data block for **36%** of the incoming data blocks that can benefit from delta compression

---

<table>
<thead>
<tr>
<th>Block 1</th>
<th>Block 2</th>
<th>Block 3</th>
</tr>
</thead>
<tbody>
<tr>
<td>SF(Block 1) = 0x7332F1CC</td>
<td>SF(Block 2) = 0x7332F1CC</td>
<td>SF(Block 3) = 0x735789CC</td>
</tr>
</tbody>
</table>

---

*Delta*
Talk Outline

- Data Reduction in Storage Systems
- DeepSketch: A New Machine Learning-based Reference Search Technique
- Evaluation Results
DeepSketch: Key Idea

- Use the **learning-to-hash method** for sketch generation
  - A promising machine learning (ML)-based approach for the nearest-neighbor search problem

![Diagram of neural network with input layer, hidden layers (HLs), and output layer.](Image)

<Learning-to-hash for content-based image retrieval>
DeepSketch: Key Idea

- Use the **learning-to-hash method** for sketch generation
  - A promising machine learning (ML)-based approach for the nearest-neighbor search problem

![Diagram of a neural network with input, hidden layers, and output layers](image)

*Image set (# of classes: C)*

*Input Layer* → *Hidden Layers (HLs)* → *Output Layer*

- *A, B = Eagle*

<Learning-to-hash for content-based image retrieval>
DeepSketch: Key Idea

- Use the learning-to-hash method for sketch generation
  - A promising machine learning (ML)-based approach for the nearest-neighbor search problem

---

**Image set (# of classes: \( C \))**

A, B = *Eagle*

**Learning-to-hash for content-based image retrieval**
DeepSketch: Overview

File System

A  B  C  X  Y  Z

Data Reduction Module

Fingerprint (FP) Generator (e.g., SHA-1)

FP Store

<table>
<thead>
<tr>
<th>Data</th>
<th>FP</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>0x32</td>
</tr>
<tr>
<td>B</td>
<td>0x47</td>
</tr>
</tbody>
</table>

Super Feature-based Sketch (SK) Generator

<table>
<thead>
<tr>
<th>Data</th>
<th>SK</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>0x73</td>
</tr>
<tr>
<td>B</td>
<td>0x9F</td>
</tr>
</tbody>
</table>

Ref. Table

<table>
<thead>
<tr>
<th>Data</th>
<th>Ref.</th>
<th>T</th>
</tr>
</thead>
<tbody>
<tr>
<td>X</td>
<td>A</td>
<td>0</td>
</tr>
<tr>
<td>Y</td>
<td>B</td>
<td>1</td>
</tr>
</tbody>
</table>

Storage Device

A  B  C  Y  Z

T: Ref. type
(0: dedup., 1: delta-comp.)
DeepSketch: Overview

File System

A  B  C  X  Y  Z  ...

Data Reduction Module

Fingerprint (FP) Generator (e.g., SHA-1)

<table>
<thead>
<tr>
<th>Data</th>
<th>FP</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>0x32</td>
</tr>
<tr>
<td>B</td>
<td>0x47</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>

FP Store

DNN-based SK generator

<table>
<thead>
<tr>
<th>Input Layer</th>
<th>Hidden Layers (HLs)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>

SK Store

<table>
<thead>
<tr>
<th>Data</th>
<th>SK</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>0x73</td>
</tr>
<tr>
<td>B</td>
<td>0xF9</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>

Ref. Table

<table>
<thead>
<tr>
<th>Data</th>
<th>Ref.</th>
<th>T</th>
</tr>
</thead>
<tbody>
<tr>
<td>X</td>
<td>A</td>
<td>0</td>
</tr>
<tr>
<td>Y</td>
<td>B</td>
<td>1</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>

T: Ref. type (0: dedup., 1: delta-comp.)

Storage Device

A  B  C  Y  Z  ...

...
DeepSketch: Challenges

- Lack of semantic information
  - Most prior learning-to-hash approaches deal with specific data types (e.g., image sets with well-defined classes)
  - DeepSketch needs to process general binary data

- Extremely high dimensional space
  - Possible bit patterns: $2^{4,096 \times 8}$ for a data block size of 4 KiB
  - Difficult to collect large enough data to train the DNN with high inference accuracy
Training the DNN of DeepSketch

Unlabeled Data Set

Clustering

Clustered Data Set

DNN Training

Clustered Data Set
(# of clusters = $C$)

Input Layer

Hidden Layers (HLs) $\ldots$ $\ldots$ $\ldots$ $\ldots$ $\ldots$

Output Layer $\{C\}$
Training the DNN of DeepSketch

Clustering

Unlabeled Data Set → Clustered Data Set

DNN Training

Clustered Data Set (# of clusters = C)

Input Layer → Hidden Layers (HLs) → Output Layer

HL1 → HL2 → ⋯ → HLN

X

a b
Training the DNN of DeepSketch

Clustering

Unlabeled Data Set

Clustered Data Set

DNN Training

Input Layer

Hidden Layers (HLs) 

Output Layer

Clustered Data Set
(# of clusters = C)

a, b ∈ X
Training the DNN of DeepSketch

Clustering

Unlabeled Data Set

Clustered Data Set

DNN Training

Clustered Data Set
(# of clusters = C)

Input Layer

Hidden Layers (HLs)

Output Layer

Generate sketches $\Rightarrow SK(a) \approx SK(b)$

a, b $\in X$
Data Clustering for DeepSketch

- Existing clustering algorithms are **unsuitable** for DeepSketch
  - K-means clustering: **No information** of appropriate initial parameter values (e.g., # of cluster $k$) in DeepSketch
  - Hierarchical clustering: Huge **computation and memory overheads** for large data sets

- Dynamic k-means clustering (DK-Clustering)
  - A version of k-means clustering that **dynamically refines** the value for $k$ while clustering a data set
  - Key idea: **Two-step** clustering that iterates
    - **Step 1:** Coarse-grained clustering to **roughly group data blocks** at low cost and remove low-impact data blocks
    - **Step 2:** Fine-grained clustering to find the **best mean block and outliers** of each group
Post-Processing for Training Data Set

- **Non-uniform distribution** of data blocks across the clusters
  - e.g., the largest 10% clusters contain 47.93% of the total data blocks.
  - Can make DNN training significantly biased towards specific data patterns

- **Resize** every cluster to have the same number of data blocks
  - If # of data blocks > \( T \) → Randomly select \( T \) data blocks
  - If # of data blocks < \( T \) → Add randomly-modified data blocks (shifting random part of data blocks)
DNN Training

- Two-step transfer learning from GreedyHash [Su+, NeurIPS’18]

1. Classification Model
   - Convolutional Layers
   - Dense Layers
   - Target Class (Cluster)
   - Transfer knowledge (learned weights)

2. Hash Network Model
   - Input Block (4 KiB)
   - Batch norm. & max pooling (K=2)
   - 8 Channels
   - K=3
   - 16 Channels
   - K=3
   - 32 Channels
   - K=3
   - Hash Layer
   - N=B
   - Target Class (Cluster)
   - N=C_{TRN}
   - N=512
   - Head Layer
   - N=C_{TRN}
   - N=4,096
   - N=512
Talk Outline

- Data Reduction in Storage Systems
- DeepSketch: A New Machine Learning-based Reference Search Technique
- Evaluation Results
Evaluation Methodology

- Compared data-reduction techniques
  - **Dedup+Comp**: Deduplication → Lossless compression (LZ4)
  - **Finesse** [Zhang+, FAST’19]
    - High-performance super-feature-based reference search
    - Deduplication → Delta compression (XDelta) → LZ4

- Workloads
  - Six workloads collected from real systems with written data
    - PC, Install, Update, Synth, Sensor, Web
    - 10% of each trace: Training data set
    - Remaining 90%: Data-reduction & performance evaluation
  - Five workloads collected while storing Stack Overflow databases (SOF)
    - **Not used for training**
    - To see the generality of DeepSketch
Overall Data-Reduction Benefits

Large data-reduction improvement: Up to 33% (21% on average)

Effective for unseen workloads (SOFs) that cannot benefit from the state-of-the-art
Higher benefits over stand-alone techniques: DeepSketch and Finesse can complement each other.

Call for future work: Significant room for improvement.
Call for future work: Non-trivial performance overheads due to approximate nearest-neighbor search (details in the full paper)
Other Analyses in the Paper

- Empirical Study on Super Feature-Based Reference Search
- Hyper-Parameter Exploration for DeepSketch’s DNN
- Performance and Space Overheads
- Reference Search Patterns of DeepSketch and Finesse
- Impact of Training Data Set
Executive Summary

- **Problem:** Existing post-deduplication delta-compression techniques provide significantly low data-reduction ratios compared to the optimal.
  - Due to the limited accuracy of reference search for delta compression
  - Cannot identify a good reference block for many incoming data blocks

- **Key Idea:** DeepSketch, a new machine learning-based reference search technique that uses the learning-to-hash method
  - Generates a given data block’s signature (sketch) using a deep neural network
  - The higher the delta-compression benefit of two data blocks, the more similar the signatures of the two blocks to each other

- **Evaluation Results:** DeepSketch reduces the amount of physically-written data
  - Up to 33% (21% on average) compared to a state-of-the-art baseline

We hope that our key ideas inspire many valuable studies going forward
P&S Modern SSDs
DeepSketch:
A New Machine Learning-Based Reference Search Technique for Post-Deduplication Delta Compression

Jisung Park\textsuperscript{1*}, Jeonggyun Kim\textsuperscript{2*}, Yeseong Kim\textsuperscript{2}, Sungjin Lee\textsuperscript{2}, and Onur Mutlu\textsuperscript{1}

\textsuperscript{1}SAFARI
\textsuperscript{2}DGVIST

ETH zürich

USENIX FAST 2022

\textsuperscript{*}J. Park and J. Kim are co-primary authors.
DeepSketch: Application Scenarios

1) Collect data for DNN training (preferably from data servers storing similar data types)

2) DNN Training (with powerful machines)

3) Update or build a new one

Data Servers

Training Data set

New Server

Trained DNN

Backup