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

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

\textsuperscript{1}SAFARI\textsuperscript{,} ETH\textsuperscript{,} zürich
\textsuperscript{2}DG\textsuperscript{,}VIST

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 \( \left( = \frac{\text{Original Data Size}}{\text{Reduced Data Size}} \right) \)
  - Deduplication \( \to \) Delta compression \( \to \) 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

A B C X ...

Data Reduction Module

Deduplication

A B C ...

Storage Device
Step 2: Delta Compression

File System

A  B  C  X  Y  ...

Data Reduction Module

A  B  C  ...

Storage Device
Step 2: Delta Compression

File System

Data Reduction Module

Storage Device

Reference

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

File System

A B C X Y Z

Data Reduction Module

Y: Delta-compressed

Storage Device

A B C Z
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

<table>
<thead>
<tr>
<th>Block 1</th>
<th>U</th>
<th>S</th>
<th>E</th>
<th>N</th>
<th>I</th>
<th>X</th>
<th>F</th>
<th>A</th>
<th>S</th>
<th>T</th>
<th>2</th>
<th>0</th>
<th>2</th>
<th>2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Feature$_1$ = $H_1$(SENI) = 0x73</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Feature$_2$ = $H_2$(FAST) = 0x32</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Feature$_3$ = $H_3$(USEN) = 0xF1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Feature$_4$ = $H_4$(S202) = 0xCC</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Block 2</th>
<th>U</th>
<th>S</th>
<th>E</th>
<th>N</th>
<th>I</th>
<th>X</th>
<th>F</th>
<th>A</th>
<th>S</th>
<th>T</th>
<th>2</th>
<th>0</th>
<th>2</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>Feature$_1$ = $H_1$(SENI) &gt; $H_1$(2020)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Feature$_2$ = $H_2$(FAST) &gt; $H_2$(2020)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Feature$_3$ = $H_3$(USEN) &gt; $H_3$(2020)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Feature$_4$ = $H_4$(S202) &gt; $H_4$(2020)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Super Feature**

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

4 KiB
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

<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 DeepSketch](image)

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

**Input Layer**

**Hidden Layers (HLs)**

**Output Layer**

C

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

![Diagram](image.png)

- **Image set** (# of classes: \( C \))
  - A
  - B

- **Input Layer**
  - \( H_L_1 \)
  - \( H_L_2 \)
  - \( \ldots \)
  - \( H_L_N \)

- **Hidden Layers (HLs)**
  - \( H_L_1 \)
  - \( H_L_2 \)
  - \( \ldots \)
  - \( H_L_N \)

- **Output Layer**
  - C

- **Generate a binary hash** \( h(X) \)
  - \( h(A) \approx h(B) \)

- <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)

```
<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>
```

### 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>0xF9</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>
```

### FP Store

```
<table>
<thead>
<tr>
<th>Ref.</th>
<th>Data</th>
<th>FP</th>
</tr>
</thead>
<tbody>
<tr>
<td>X</td>
<td>A</td>
<td>0x32</td>
</tr>
<tr>
<td>Y</td>
<td>B</td>
<td>0x47</td>
</tr>
<tr>
<td>T</td>
<td></td>
<td>...</td>
</tr>
</tbody>
</table>
```

### SK Store

```
<table>
<thead>
<tr>
<th>Ref.</th>
<th>Data</th>
<th>SK</th>
</tr>
</thead>
<tbody>
<tr>
<td>X</td>
<td>A</td>
<td>0x73</td>
</tr>
<tr>
<td>Y</td>
<td>B</td>
<td>0xF9</td>
</tr>
<tr>
<td>T</td>
<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.)
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>
<tr>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>

DNN-based SK generator

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: $2^{4,096\times8}$ 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

- **Clustering**
  - Unlabeled Data Set
  - Clustered Data Set

- **DNN Training**
  - Clustered Data Set (# of clusters = \( C \))
  - Input Layer
  - Hidden Layers (HLs): \( HL_1, HL_2, \ldots, HL_N \)
  - 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

C

HL1

HL2

⋯

HLN

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

HL_1

HL_2

⋯

HL_N

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]
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 w/ 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.
Performance Overhead

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**, Jeonggyun Kim, Yeseong Kim, Sungjin Lee, and Onur Mutlu

1 SAFARI
ETH Zürich

2 DGVIST
Daegu Gyeongbuk Institute of Science & Technology

USENIX FAST 2022

*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

Backup