Hashing method for Image Retrieval

Student: Huy Le-Nhat

Instructor: Tien-Dung Mai

University of Information Technology

Subject: CS530 - Computer Science

Email: 20520056@gm.uit.edu.vn

Outline

Outline

1. Purpose
2. Introduction
3. Related Works
4. Approach
5. Experiment
6. Conclusion

Purpose

- What hashing for image retrieval is
- Read the article *Deep Supervised Hashing for Fast Image Retrieval* [1] CVPR 2016
- Learn techniques (theory & experiment):
  - Deep Supervised Hashing (DSH) [3]
  - Locally Sensitive Hashing (LSH) [2]
  - Compare DSH & LSH

Outline

Image Retrieval Problem

Problem Description

- Input: A query image
- Output: A set of retrieved images that are similar to the query image

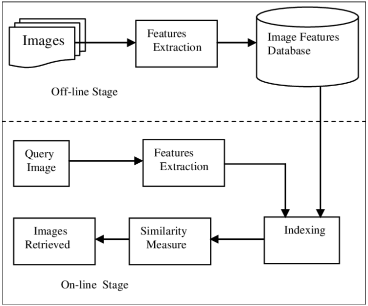

Figure 1: Image retrieval - flow chart

Some index & searching methods for image retrieval:

- Text-based method
- Feature-Based method
- **Hashing-Based method**
- Graph-Based method
- Learning-Based method
- Hybrid method

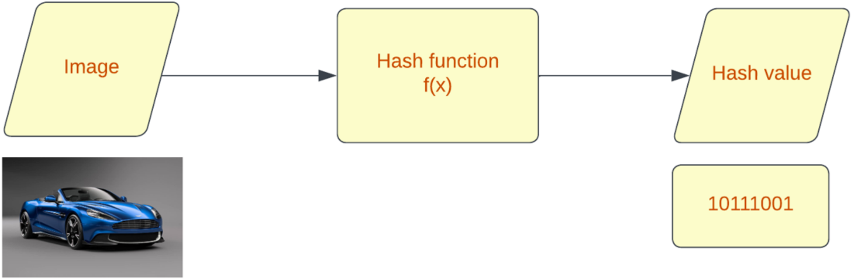

Figure 2: Image hashing

Important: More similar images $\longleftrightarrow$ more similar hash values

Hashing Method in Image Retrieval

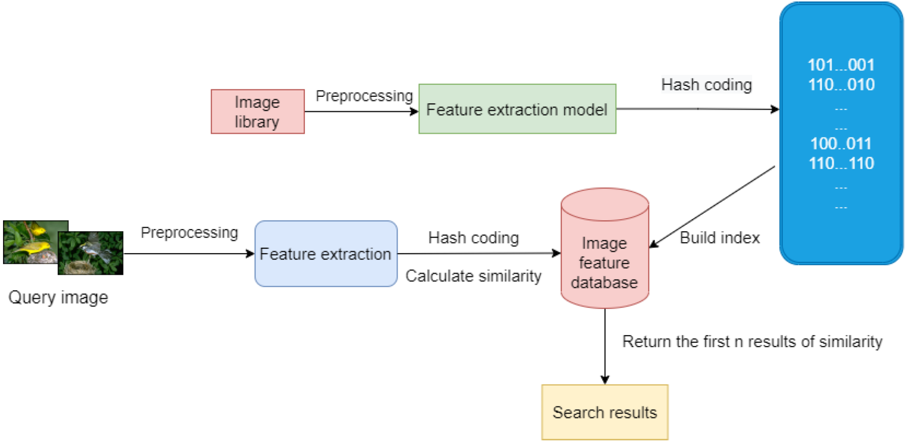

Figure 3: Flow chart of the image retrieval based on hashing method

Hashing Method in Image Retrieval

Properties

- Advantages: compact and fast retrieval time
- Disadvantages: low-dimensional hash values $\longrightarrow$ contain less information

Hashing is useful for tasks like efficient data storage and retrieval, especially for big data (images).

Some hashing methods:

- **Locally Sensitive Hashing**
- Iterative Quantization
- Spectral Hashing
- **Deep Supervised Hashing**

Outline

Concept: hash similar input items into the same buckets

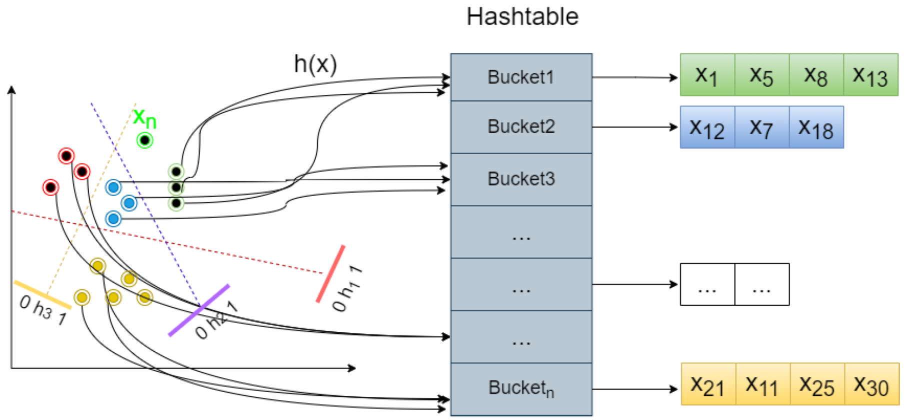

Figure 4: Locally Sensitive Hashing

Random Projection Method & Multi-hash Table

$$\begin{bmatrix} \quad Projected(P) \quad \end{bmatrix}_{k\times n} = \begin{bmatrix} \quad Random(R) \quad \end{bmatrix}_{k\times d} \begin{bmatrix} \quad Original(D) \quad \end{bmatrix}_{d\times n}$$

Figure 5: Random Projection Method

- One random projection is respective to one hash table
- Multi-hash table: more accuracy
- Calculate similarity: Jaccard distance, Cosine distance, Hamming distance, Euclidean distance

Locally Sensitive Hashing (LSH)

- Simple and quick to implement, fast query
- Well-suited for approximate visual-similarity search in large datasets
- Disadvantages: Require a long-length hash code for high accuracy, inefficiently preserve images' semantics

Deep Supervised Hashing (DSH)

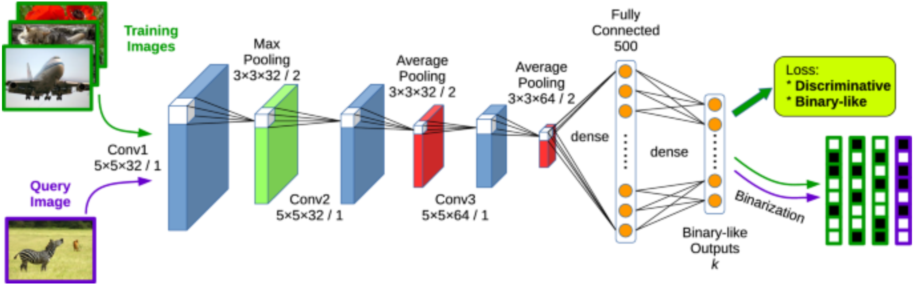

Figure 6: Deep Supervised Hashing Example

Deep Supervised Hashing (DSH)

- Well-preserved semantic information
- High accuracy and fast query
- Disadvantages: Complex in network design, parameter tuning, and needing enough labeled data to work effectively

Outline

Pipeline

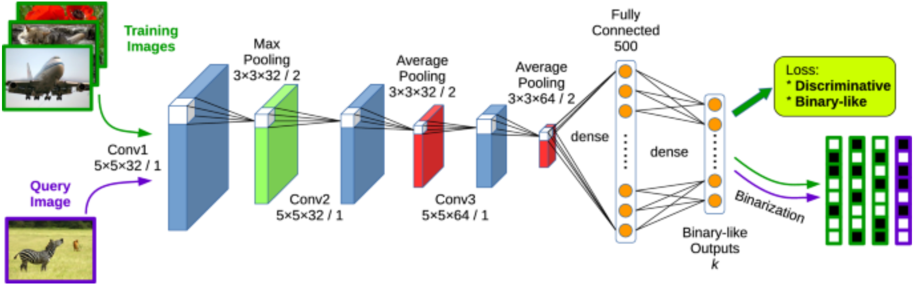

Figure 7: Model pipeline

Utilize the Siamese structure to pass sequentially each pair of images to the training process.

Loss Function

Equation

$$
\begin{aligned}
L(b_1, b_2, y) = & \frac{1}{2}\,(1 - y)\, D_h(b_1, b_2) \\
& + \frac{1}{2}\, y\, max(m - D_h(b_1, b_2), 0)
\end{aligned} \tag{1}
$$

$$s.t. b_j \in \{+1, -1\}^k, j \in \{1, 2\}$$

Where:

- b_i is a binary vector
- $y = 0$ if two binary vectors are similar, and $y = 1$ otherwise
- $D_h(.,.)$ denotes the Hamming distance between two binary vectors
- $m > 0$ is a margin threshold parameter

Loss Function

Total Loss Function

$$\mathbf{L} = \sum_{i=1}^{N} L(b_{i,1}, b_{i,2}, y) \tag{2}$$
$$s.t. b_{i,j} \in \{+1, -1\}^k, i \in \{1..N\}, j \in \{1, 2\}$$

Issue: intractable to train the network with back propagation algorithm because of the binary constraints on $b_{i,j} \longrightarrow$ Relaxation

Relaxation

Relax constraints:

- $b_{i,j} \longrightarrow$ real value
- Hamming distance $\longrightarrow$ Euclidean distance

Loss Function

$$
\begin{aligned}
L(b_1, b_2, y) =& \frac{1}{2}(1-y)\,\|b_1 - b_2\|_2^2 \\
&+ \frac{1}{2}\,y\,max(m - \|b_1 - b_2\|_2^2, 0)
\end{aligned} \tag{3}
$$

Regularization

Regularizer

$$\alpha \left(\| \, |b_1| - 1 \, \|_1 + \| \, |b_2| - 1 \, \|_1 \right) \tag{4}$$

Function: encourage the real-valued outputs to approach the desired discrete values (reducing the discrepancy between the real-valued network output space and the desired Hamming space)

Regularization

Loss Function

$$
\begin{aligned}
L(b_1, b_2, y) = & \frac{1}{2}(1-y)\,\|b_1 - b_2\|_2^2 \\
& + \frac{1}{2}\,y\,max(m - \|b_1 - b_2\|_2^2, 0) \\
& + \alpha\,(\|\,|b_1| - 1\,\|_1 + \|\,|b_2| - 1\,\|_1)
\end{aligned} \tag{5}
$$

Total Loss Function

$$
\begin{aligned}
\mathbf{L} = \sum_{i=1}^{N} \{ & \frac{1}{2}(1-y_i)\,\|b_{i,1} - b_{i,2}\|_2^2 \\
& + \frac{1}{2}\,y_i\,max(m - \|b_{i,1} - b_{i,2}\|_2^2, 0) \\
& + \alpha\,(\|\,|b_{i,1}| - 1\,\|_1 + \|\,|b_{i,2}| - 1\,\|_1)\}
\end{aligned} \tag{6}
$$

Loss Function

Sub-gradients

$$
\begin{aligned}
\frac{\partial Term_1}{\partial b_{i,j}} &= (-1)^{j+1}\left(1-y_i\right)\left(b_{i,1}-b_{i,2}\right) \\
\frac{\partial Term_2}{\partial b_{i,j}} &= \begin{cases} (-1)^{j}\, y_i \left(b_{i,1}-b_{i,2}\right) & , \|b_{i,1}-b_{i,2}\|_2^2 < m \\ 0 & , \text{otherwise} \end{cases} \\
\frac{\partial Regularizer}{\partial b_{i,j}} &= \begin{cases} 1 & , -1 \le b_{i,j} \le 0 \text{ } or \text{ } b_{i,j} \ge 1 \\ -1 & , \text{otherwise} \end{cases}
\end{aligned} \tag{7}
$$

Outline

Experiment

- Dataset: Fashion-MNIST (10 classes, 70000 images)
- Hyper-parameters:
  - DSH:
    - learning rate: 0.003
    - hash-size: 8, 12 and 16
    - margin (m): hash-size * 2
    - regularizer (α): 0.005 and 0.1
    - epoch: 20
  - LSH:
    - feature extractor: HOG (Histogram of oriented gradients)
    - hash-size: 8, 12 and 16
    - number of hash tables: 8 and 16
- Evaluation metric: $mAP@10$

Experiment

No. Hash-tables \Hash-size	8-bits	12-bit	16-bits
8	0.6223	0.6401	0.6494
16	0.6775	0.6923	0.6940

Table 1: LSH - *mAP*@10

Regularizer \Hash-size	8-bits	12-bit	16-bits
0.005	0.7305	0.7650	0.7744
0.01	0.7184	0.7576	0.7801

Table 2: DSH - *mAP*@10

Experiment

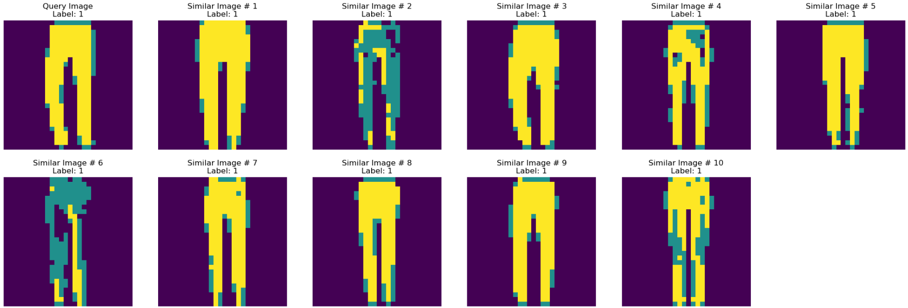

Figure 8: Visualization 1

Experiment

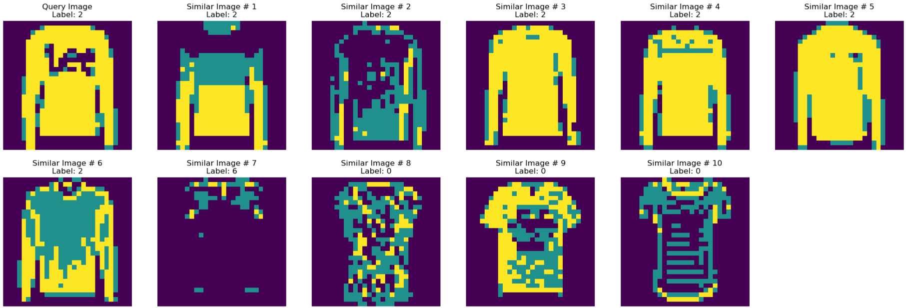

Figure 9: Visualization 2

Outline

Conclusion

Hashing method: compact and fast retrieval time

DSH:

- Using labeled data to train the deep neural network
- Higher precision than many other hashing methods: LSH, SH (Spectral Hashing), ITQ (Interactive Quantization), CNNH, ... (proven in the article)

Future works: build a better structure Deep Neural Network for DSH

Thanks for listening

Any Questions? Comment?

References

[1] Haomiao Liu et al. "Deep supervised hashing for fast image retrieval". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2016, pp. 2064–2072.

[2] Youmeng Luo et al. "Image Retrieval Algorithm Based on Locality-Sensitive Hash Using Convolutional Neural Network and Attention Mechanism". In: *Information* 13.10 (2022), p. 446.

[3] Xiaopeng Zhang. "A survey on deep hashing for image retrieval". In: *arXiv preprint arXiv:2006.05627* (2020).