Supervised Discrete Hashing

Fumin Shen\textsuperscript{1}, Chunhua Shen\textsuperscript{2}, Wei Liu\textsuperscript{1}, Heng Tao Shen\textsuperscript{4}
\textsuperscript{1}University of Electronic Science and Technology of China. \textsuperscript{2}University of Adelaide; and Australian Centre for Robotic Vision. \textsuperscript{3}IBM Research. \textsuperscript{4}The University of Queensland.

Recently, learning based hashing techniques have attracted broad research interests due to the resulting efficient storage and retrieval of images, videos, documents, etc. However, a major difficulty of learning to hash lies in handling the discrete constraints imposed on the needed hash codes. In general, the discrete constraints imposed on the binary codes that the target hash functions generate lead to mixed-integer optimization problems—which is generally NP hard. To simplify the optimization involved in a binary code learning procedure, most of the aforementioned methods choose to first solve a relaxed problem through directly discarding the discrete constraints, and then threshold the continuous outputs to be binary. This greatly simplifies the optimization but, unfortunately, the approximated solution is typically of low quality and often makes the final hash functions less effective, possibly due to the accumulated quantization errors. This is especially the case when long-length codes are needed.

Directly learning the binary codes without relaxations would be preferred if (and only if) a tractable and scalable solver is available. The importance of discrete optimization in hashing has been rarely taken into account when long-length codes are needed. This is especially the case when long-length codes are needed.

Supervised Discrete Hashing (SDH) is the discrete cyclic coordinate descent (DCC) method. In other words, We learn \( b \) bit by bit. Let \( z^l \) be the \( l \)th row of \( B \), \( l = 1, \ldots, L \) and \( B \) the matrix of \( B \) excluding \( z \). Then \( z \) is one bit for all \( n \) samples. Similarly, let \( q^l \) be the \( l \)th row of \( Q \), \( Q' \) the matrix of \( Q \) excluding \( q \), \( q' \) the \( l \)th row of \( W \) and \( W' \) the matrix of \( W \) excluding \( v \). Then we have w.r.t. \( z \):

\[
\begin{aligned}
\min_{z} & \quad (v^T W'^T b - q^T)z \\
\text{s.t.} & \quad z \in \{-1,1\}^n.
\end{aligned}
\]

This problem has the optimal solution

\[
z = \text{sgn}(q - B' W' v).
\]

By carefully choosing loss functions of the classifier, the DCC method produces the optimal hash bits in a closed form, which consequently makes the entire optimization procedure very efficient. Therefore, the proposed binary code learning method can easily deal with large-scale datasets. We name the proposed supervised hashing method employing discrete cyclic coordinate as Supervised Discrete Hashing (SDH).

**Discrete or Not?** To see how much the discrete optimization will benefit the hash code learning, we perform a comparison of our hash formulation (1) with or without the discrete constraints. The comparative results on CIFAR are shown in Table 1. As can be seen, our discrete hashing framework SDH consistently yields better hash codes than the relaxed one by removing the sign function. In particular for precision, the performance gaps between these two methods are increased with longer hash bits.

<table>
<thead>
<tr>
<th>Method</th>
<th>Precision</th>
<th>MAP</th>
<th>Training time</th>
<th>Test time</th>
</tr>
</thead>
<tbody>
<tr>
<td>BRE</td>
<td>0.1299</td>
<td>0.1156</td>
<td>12042.0</td>
<td>6.4e-5</td>
</tr>
<tr>
<td>MLH</td>
<td>0.2251</td>
<td>0.1730</td>
<td>2297.5</td>
<td>3.2e-5</td>
</tr>
<tr>
<td>KSH</td>
<td>0.1656</td>
<td>0.3822</td>
<td>2625.0</td>
<td>3.1e-6</td>
</tr>
<tr>
<td>SSH</td>
<td>0.2860</td>
<td>0.2091</td>
<td>96.9</td>
<td>3.6e-6</td>
</tr>
<tr>
<td>CCA-ITQ</td>
<td>0.3524</td>
<td>0.3379</td>
<td>4.3</td>
<td>1.7e-6</td>
</tr>
<tr>
<td>FastHash</td>
<td>0.1880</td>
<td>0.4187</td>
<td>1340.7</td>
<td>7.1e-4</td>
</tr>
<tr>
<td>SDH</td>
<td>0.4229</td>
<td>0.4555</td>
<td>62.6</td>
<td>2.6e-6</td>
</tr>
</tbody>
</table>

Table 1: Comparative results of our method with discrete constraints or relaxed ones.

The efficacy of SDH is validated by the superior results over several state-of-the-arts on CIFAR with 64 bits. The code for SDH is available at https://github.com/bd622/DiscretHashing.

---