Supplementary Material for
“Bayes Merging of Multiple Vocabularies for Scalable Image Retrieval”

1. Overview

This document includes supplementary material to “Bayes Merging of Multiple Vocabularies for Scalable Image Retrieval”. Included are the fast versions of Bayes merging and some sample retrieval results.

2. Fast Implementation of Bayes Merging

To speed up Bayes merging, besides the offline computation of the cardinality ratio $\frac{\text{Card}(A \cap B)}{\text{Card}(A \cup B)}$, the online process is presented in Algorithm 2 and Algorithm 1 (For one query feature).

These two algorithms correspond to two alternatives of inverted index. In Algorithm 1, the inverted list stores one entry per descriptor, which is required by HE [1, 2]. On the other hand, in Algorithm 2, the inverted index stores one entry per image [4], where the image identifier and the TF value are stored. For “Bayes merging + HE” method, the implementation is essentially built on Algorithm 1; for “Bayes merging” alone, Algorithm 2 is employed.

Here, we illustrate the case of Bayes merging of two vocabularies, because the pseudo-code does not look too long, and because the performance of merging two vocabularies is very close to multiple ones. Note that Bayes merging of multiple ($K \geq 3$) vocabularies shares essentially the same procedure.

Consequently, given two sets of indexed features $A$ and $B$, the Bayes merging method has the same computation complexity $O(\text{card}(A) + \text{card}(B))$ with baseline $B_1$. In other words, for each query feature, we only have to traverse the lists of indexed features once.

3. Sample Retrieval Results

In this supplementary material, we also provide some sample retrieval results on the Holidays [1], Oxford [4], and Ukbench datasets [3]. These results are obtained using Bayes merging of two vocabularies of size 20K. Note that HE is not employed here. See Fig. 1, Fig. 2, and Fig. 3, respectively.

---

Algorithm 1 Bayes merging for one entry per descriptor

**Input:**
- two arrays of the image indices $idx_1$, $idx_2$;
- Arrays lengths $\text{len}(idx_1) = \text{len}_1$, $\text{len}(idx_2) = \text{len}_2$;
- Two indicators $i \leftarrow 0$, $j \leftarrow 0$;
- Initial scores of the images $s$;
- Bayes weight $w$;

**Iteration:**
1: while $i < \text{len}_1$ and $i < \text{len}_2$ do
2:  if $idx_1[i] < idx_2[j]$ then
3:    $s[\text{idx}_1[i]] \leftarrow s[idx_1[i]] + 1$;
4:    $i \leftarrow i + 1$;
5:  else
6:    if $idx_1[i] > idx_2[j]$ then
7:      $s[\text{idx}_2[j]] \leftarrow s[idx_2[j]] + 1$;
8:      $j \leftarrow j + 1$;
9:    else
10:      do
11:        $s[\text{idx}_1[i]] \leftarrow s[idx_1[i]] + w$;
12:        $i \leftarrow i + 1$;
13:      while $i < \text{len}_1$ and $idx_1[i] == \text{id}_x_1[i-1]$  
14:      do
15:        $s[\text{idx}_2[j]] \leftarrow s[idx_2[j]] + w$;
16:        $j \leftarrow j + 1$;
17:      while $j < \text{len}_2$ and $idx_2[j] == \text{id}_x_2[j-1]$  
18:    end if
19:  end if
20: end while
21: while $i < \text{len}_1$ do
22:  $s[\text{idx}_1[i]] \leftarrow s[idx_1[i]] + w$;
23: end while
24: while $j < \text{len}_2$ do
25:  $s[\text{idx}_2[j]] \leftarrow s[idx_2[j]] + w$;
26: end while

**Output:**
The updated score $s$. 

1
Algorithm 2 Bayes merging for one entry per image

Input:
- two arrays of the image indices $idx_1, idx_2$
- two arrays of the TF values $tf_1, tf_2$
- Arrays lengths $len(idx_1) = len_1, len(idx_2) = len_2$
- Two indicators $i \leftarrow 0, j \leftarrow 0$
- Initial scores of the images $s$
- Bayes weight $w$

Iteration:
1: while $i < len_1$ and $i < len_2$ do
2:   if $idx_1[i] < idx_2[j]$ then
3:     $s[idx_1[i]] \leftarrow s[idx_1[i]] + tf_1[i];$
4:     $i \leftarrow i + 1;$
5:   else
6:     if $idx_1[i] > idx_2[j]$ then
7:       $s[idx_2[j]] \leftarrow s[idx_2[j]] + tf_2[j];$
8:       $j \leftarrow j + 1;$
9:       else
10:      $s[idx_1[i]] \leftarrow s[idx_1[i]] + w \cdot tf_1[i];$
11:     $i \leftarrow i + 1;$
12:      $s[idx_2[j]] \leftarrow s[idx_2[j]] + w \cdot tf_2[j];$
13:      $j \leftarrow j + 1;$
14:   end if
15: end if
16: end while
17: while $i < len_1$ do
18:   $s[idx_1[i]] \leftarrow s[idx_1[i]] + w \cdot tf_1[i];$
19: end while
20: while $j < len_2$ do
21:   $s[idx_2[j]] \leftarrow s[idx_2[j]] + w \cdot tf_2[j];$
22: end while

Output:
The updated score $s$. 
Figure 1. Sample retrieval results on Holidays dataset. For each query (left), retrieval results of four methods are presented in each row, i.e., baseline $B_0$ using vocabulary 1 and 2 ($B_{0,1}$ and $B_{0,2}$, respectively), baseline $B_1$, and the proposed Bayes merging ($\text{Bayes}$). The images start from the second one in the rank list. The ground truth images are in green boxes.
Figure 2. Sample retrieval results on Ukbench dataset. For each query (left), retrieval results of four methods are presented in each row, i.e., baseline B0 using vocabulary 1 and 2 (B0_1 and B0_2, respectively), baseline B1, and the proposed Bayes merging (Bayes). The images start from the second one in the rank list. The ground truth images are in green boxes.
Figure 3. Sample retrieval results on Oxford dataset. For each query (left), retrieval results of four methods are presented in each row, i.e., baseline B₀ using vocabulary 1 and 2 (B₀,₁ and B₀,₂, respectively), baseline B₁, and the proposed Bayes merging (Bayes). For the first and second queries, the images start from the second one in the rank list; for the third query, the images start from the 11th in the rank list. The ground truth images are in green boxes.
References


