Learning with Average Precision: Training Image Retrieval with a Listwise Loss

Jérôme Revaud Jon Almazán Rafael S. Rezende César Roberto de Souza

NAVER LABS Europe

1. Backpropagation of mAP_Q

In this section, we expand upon Section 3.2 of our submission to further explain the backpropagation process for a batch of image descriptors $D = [d_1, \dots, d_B]$ and groundtruth $Y = [Y_1, \dots, Y_B]$. Additionally, we will demonstrate the differentiability of the loss function L. We separate the algorithm in three stages, accordingly to Fig. 3 of our submission.

Stage 1. This stage consists only of a forward pass of our network in evaluation mode, *i.e.* no backpropagation is needed. For each image I_i of the batch, its descriptor d_i is computed as $d_i = f_\Theta(I_i)$ and the matrix D is determined by the stacking of these descriptors by the rows.

Stage 2. In this stage, we compute the loss $\ell = L(D, Y)$ given by:

$$\ell = 1 - \frac{1}{B}\sum_{q=1}^{B}\sum_{m=1}^{M} \hat{P}_m(d_q^\top D, Y_q)\ \Delta\hat{r}_m(d_q^\top D, Y_q). \quad (1)$$

Since, at this stage, the backpropagation method stops at the descriptors level, our goal is to compute the derivatives $\frac{\partial \ell}{\partial D} = \left(\frac{\partial \ell}{\partial d_i}\right)_{1\le i\le B}$. We introduce some mathematical notations to ease the readability of this section. Let $S^q = d_q^\top D$ in $[-1, 1]^B$ denote the similarities of the batch images with a query I_q in the batch, $\delta_m^q = \delta(S^q, m)$ in $[0, 1]^B$ denote the soft-assignment of S^q to the m-th bin and $\xi_m^q = \sum_{k=1}^m \delta_k^q$ in $\mathbb{R}_+^B$ denotes the sum of all the soft-assignments of S^q up until the m-th bin. Moreover, $\hat{P}_m^q = \hat{P}_m(S^q, Y_q)$ and $\Delta\hat{r}_m^q = \Delta\hat{r}_m(S^q, Y_q)$ denote the precision and incremental recall at bin m of S^q, respectively. These notations allow us to rewrite $\hat{P}_m^q$ and $\Delta\hat{r}_m^q$ as:

$$\hat{P}_m^q = \frac{(\xi_m^q)^\top Y_q}{(\xi_m^q)^\top \mathbf{1}}, \quad (2)$$

$$\Delta\hat{r}_m^q = \frac{(\delta_m^q)^\top Y_q}{N^q}. \quad (3)$$

We now derivate Eq. (1) w.r.t. d_i and, by applying the chain-rule, we obtain:

$$\frac{\partial \ell}{\partial d_i} = -\frac{1}{B}\sum_{q=1}^{B}\sum_{m=1}^{M}\left(\frac{\partial \hat{P}_m^q}{\partial d_i}\Delta\hat{r}_m^q + \hat{P}_m^q\frac{\partial \Delta\hat{r}_m^q}{\partial d_i}\right) \quad (4)$$

$$= -\frac{1}{B}\sum_{q=1}^{B}\sum_{m=1}^{M}\left(\Delta\hat{r}_m^q\frac{\partial \hat{P}_m^q}{\partial \xi_m^q}\frac{\partial \xi_m^q}{\partial S^q} + \hat{P}_m^q\frac{\partial \Delta\hat{r}_m^q}{\partial \delta_m^q}\frac{\partial \delta_m^q}{\partial S^q}\right)\frac{\partial S^q}{\partial d_i}. \quad (5)$$

We can separately compute each of the terms of Eq. (5). The derivatives $\frac{\partial \delta_m^q}{\partial S^q}$ and $\frac{\partial \xi_m^q}{\partial S^q}$ in $\mathbb{R}^{B\times B}$ are defined and can be backpropagated thanks to the definition of the triangular kernel:

$$\left(\frac{\partial \xi_m^q}{\partial S^q}\right)_{ij} = \left(\sum_{k=1}^{m}\frac{\partial \delta_k^q}{\partial S^q}\right)_{ij}, \quad (6)$$

$$\left(\frac{\partial \delta_m^q}{\partial S^q}\right)_{ij} = -\frac{\text{sign}(S_i^q - b_m)}{\Delta}\mathbb{1}\left[|S_i^q - b_m| \le \Delta\right]\mathbb{1}[i = j]. \quad (7)$$

Next, the missing derivatives, $\frac{\partial \hat{P}_m^q}{\partial \xi_m^q}$ and $\frac{\partial \Delta\hat{r}_m^q}{\partial \delta_m^q}$ in $\mathbb{R}^{1\times B}$ and $\frac{\partial S^q}{\partial d_i}$ in $\mathbb{R}^{B\times D}$ can be expressed as follows:

$$\frac{\partial \hat{P}_m^q}{\partial \xi_m^q} = \frac{1}{((\xi_m^q)^\top \mathbf{1})^2}(\xi_m^q)^\top\left(\mathbf{1}Y_q^\top - Y_q\mathbf{1}^\top\right), \quad (8)$$

$$\frac{\partial \Delta\hat{r}_m^q}{\partial \delta_k^q} = \frac{1}{N^q}Y_q^\top, \quad (9)$$

$$\left(\frac{\partial S^q}{\partial d_i}\right)_j = d_j^\top\mathbb{1}[i = q] + d_q^\top\mathbb{1}[i = j]. \quad (10)$$

And finally, putting together Eqs. (6), (7), (8), (9), (10) in Eq. (5) proves that L is differentiable and its derivative provides useful information for backpropagation.

Stage 3. In the last stage, the goal is to compute $\frac{\partial \ell}{\partial \Theta}$. For each i in $[1, \dots, B]$, $d_i = f_\Theta(I_i)$ and $\frac{\partial d_i}{\partial \Theta} = \frac{\partial}{\partial \Theta} f_\Theta(I_i)$ are computed according to the network backpropagation algorithm, and $\frac{\partial \ell}{\partial d_i}$ is obtained in stage 2. Therefore, the chain-rule give us:

$$\frac{\partial \ell}{\partial \Theta} = \sum_{i=1}^{B} \frac{\partial \ell}{\partial d_i} \frac{\partial d_i}{\partial \Theta}. \tag{11}$$

2. AP_Q is a good approximation of the AP

In this section, we illustrate the quality of the approximation given by AP_Q to the true AP. In order to assess this approximation, we scatter plot the true AP and AP_Q for all queries of the $\mathcal{R}$Paris dataset in Fig. 1. We observe a correlation of ~ 0.98 between them.

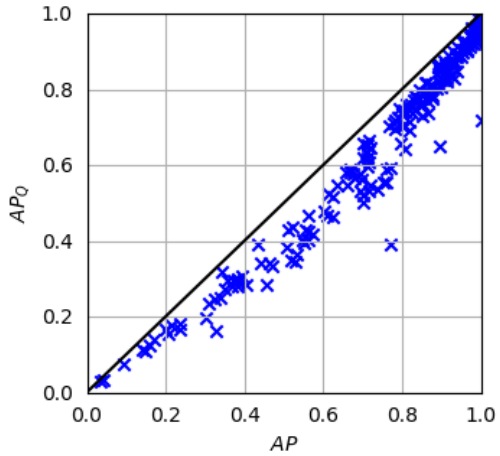

Figure 1. Scatter plot of the true AP versus AP_Q for $\mathcal{R}$Paris [2].

3. Hyperparameters

In this section, we present additional information regarding the Table 1 in our original submission. In Table 1 above we give an explicit list of the hyperparameters used by each method. As it can be seen, our approach requires much fewer hyperparameters than [1, 3]. This means much less effort is required during training since there are less variables to tune and to optimize using costly procedures such as grid-searches and cross-validation.

Specifically, in order to use hard negative mining (HNM) as in [1], one would need to find correct values for at least 6 hyperparameters: the number N of images in mining pool; the n for when selecting the top-n hardest triplets for mining; the number k of times before updating good candidates; the number bs of triplets before an update; the margin m of the loss; the batch size b. Likewise, using the method of [3] requires determining values for at least 7 hyperparameters: the margin τ of the loss, the size k^* of the pool of candidate positives, the overlap threshold t_i, the scale-change threshold t_s, the number n_h of times hard negatives are re-mined, the number n_n of negative images in the tuple, and the batch size b. Choosing a wrong value for any of these parameters can significantly impact training and lead to poorer results.

Table 1. Hyper-parameters for different methods

Method	Hyper-parameters (besides optimizer)	Total
R-MAC (TL) [1]	N, bs, k, b, m, n	6
GeM (CL) [3]	n_n, k^*, t_i, t_s, n_h, b, τ, ...	7+
GeM (AP) [ours]	n_b, b	2

Abbreviations: N, images in mining pool; n, top-n hardest triplets for hard negative mining (HNM); k, number of times before updating good candidates; bs, number of triplets before an update; m, τ, margin of the loss; b, batch size; k^*, pool of candidate positives; t_i, overlap threshold; t_s, scale-change threshold; n_h, number of times hard negatives are re-mined; n_n, number of negative images in the tuple; n_b, number of bins.

In contrast, our loss only requires 2 hyper-parameters which have safe defaults that are not very sensitive to changes (please refer to Figures 4 and 5 in our submission). Those are the number of bins n_b, and the batch size b.

4. Multistaged backpropagation algorithm

In this section, we provide in Algorithm 1 a pseudocode for our multistaged backpropagation, as presented in Section 3.3 of our submission. The code is split similarly to Figure 2 of our submission in order to make it easier to follow.

Algorithm 1: Efficient training for any batch size.

Input: Training batch $\mathcal{B} = \{(I_1, y_1), \dots, (I_B, y_B)\}$
begin

```
# Stage 1
for i in 1..B; do
    compute d_i ← f_Θ(I_i)

# Stage 2
create D ← [d_1, ..., d_B] ∈ ℝ^{B×C}
create Y ∈ {0,1}^{B×B} such that Y_ij = 𝟙[y_i = y_j]
ℓ ← L(D, Y) # compute the loss
compute ∂ℓ/∂D ∈ ℝ^{B×C}

# Stage 3
init ∂ℓ/∂Θ ← 0
for i in 1..B; do
    compute d_i ← f_Θ(I_i)
    compute ∂d_i/∂Θ ← ∂/∂Θ f_Θ(I_i)
    accumulate ∂ℓ/∂Θ ← ∂ℓ/∂Θ + ∂d_i/∂Θ (∂ℓ/∂D)_i # w/chain-rule
update Θ ← Θ − γ ∂ℓ/∂Θ # optimizer step
```

end

5. Qualitative results

In this section, we provide additional examples of queries and their respective top-10 result images as obtained by our retrieval model (referred as "GeM (AP)" in Table 1 of our submission) in the $\mathcal{R}$Paris (Table 2) and $\mathcal{R}$Oxford (Table **??**) datasets [2]. In these tables, each retrieved image is marked with a colored border indicating whether they belong to the **positive** or **negative** groups of the *medium* benchmark associated with these datasets (see Section 4.1 of our submission).

Failure cases. At the bottom of these tables, we also list the three worst performing queries for each dataset. Those queries serve as demonstrations of failure cases that can be used to indicate in which situations our models could be improved. As it can be seen, correctly retrieving images belonging to these queries requires a significant level of attention to details, such as subtle differences in adornment statues (see last rows of Tables 2 and **??**) or similarly looking building façades seen from a short distance (see last rows of Table **??**).

Hard benchmark. We also include two other tables (Tables **??** and **??**) with exactly the same queries as Tables 2 and **??**, but showing their evaluation under the *hard* benchmark of $\mathcal{R}$Paris and $\mathcal{R}$Oxford. In this protocol, we remove easy-to-retrieve images, focusing instead on difficult cases. While this results in a higher amount of mistakes, our model still manages to correctly retrieve images with varying lighting conditions (see Table **??**) and high levels of occlusion (see 2$^{\text{nd}}$ row of Table **??** and 5$^{\text{th}}$ row of Table **??**).

References

[1] A. Gordo, J. Almazán, J. Revaud, and D. Larlus. End-to-end learning of deep visual representations for image retrieval. *IJCV*, 2017. 3, 1

[2] F. Radenović, A. Iscen, G. Tolias, Y. Avrithis, and O. Chum. Revisiting Oxford and Paris: Large-scale image retrieval benchmarking. In *CVPR*, 2018. 1, 5, 2

[3] F. Radenović, G. Tolias, and O. Chum. Fine-tuning CNN image retrieval with no human annotation. *TPAMI*, 2018. 3, 1

Table 2. Top-10 results for 5 random queries and 3 worst-performing queries in $\mathcal{R}$Paris [2].

Query	Top-1	Top-2	Top-3	Top-4	Top-5	Top-6	Top-7	Top-8	Top-9	Top-10
Random queries										
Top failure cases