Hierarchical Average Precision Training for Pertinent Image Retrieval – Supplementary Material –

A Method

A.1 $\mathcal{H}$-rank

We define the $\mathcal{H}$-rank in the main paper as:

$$\mathcal{H}\text{-rank}(k) = \text{rel}(k) + \sum_{j \in \Omega^+} \min(\text{rel}(k), \text{rel}(j)) \cdot H(s_j - s_k) \ . \tag{1}$$

We detail in Fig. 1 how the $\mathcal{H}$-rank in Eq. (1) is computed in the example from Fig. 2b of the main paper. Given a "Lada #2" query, we set the relevances as follows: if $k \in \Omega^{(3)}$ (*i.e.* k is also a "Lada #2"), $\text{rel}(k) = 1$; if $k \in \Omega^{(2)}$ (*i.e.* k is another model of "Lada"), $\text{rel}(k) = 2/3$; and if $k \in \Omega^{(1)}$ (k is a "Car"), $\text{rel}(k) = 1/3$. Relevance of negatives (other vehicles) is set to 0.

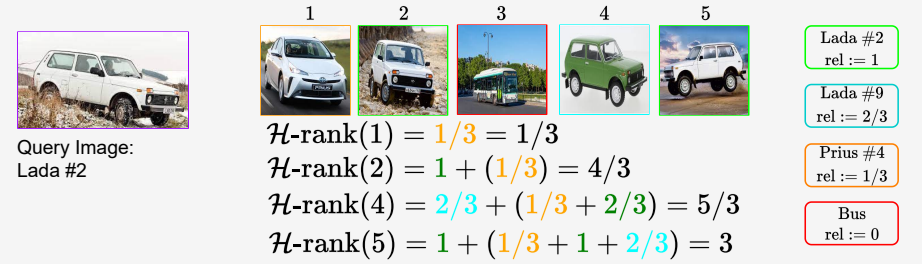


Fig. 1: $\mathcal{H}$-rank for each retrieval results given a "Lada #2" query with relevances of Sec. A.1 and the hierarchical tree of Fig. 2a of the main paper.

In this instance, $\mathcal{H}\text{-rank}(2) = 4/3$ because $\text{rel}(2) = 1$ and $\min(\text{rel}(1), \text{rel}(2)) = \text{rel}(1) = 1/3$. Here, the closest common ancestor in the hierarchical tree shared by the query and instances 1 and 2 is "Cars". For binary labels, we would have $\text{rank}^+(2) = 1$; this would not take into account the semantic similarity between the query and instance 1.

A.2 $\mathcal{H}$-AP

We define $\mathcal{H}$-AP in the main paper as:

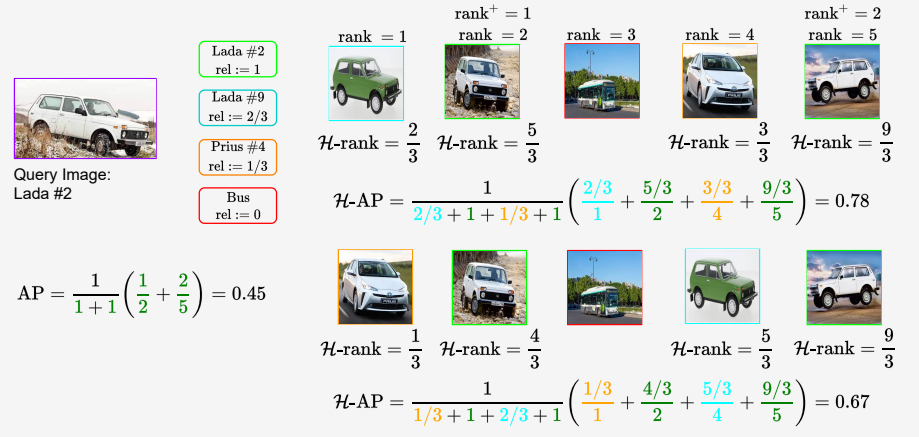


Fig. 2: AP and $\mathcal{H}$-AP for two different rankings when Given a "Lada #2" query and relevances of Sec. A.1. The $\mathcal{H}$-AP of the top row is greater (0.78) than the bottom one's (0.67) as the error in rank = 1 is less severe for the top row. Whereas the AP is the same for both rankings (0.45).

$$\mathcal{H}\text{-AP} = \frac{1}{\sum_{k\in\Omega^+}\text{rel}(k)} \sum_{k\in\Omega^+} \frac{\mathcal{H}\text{-rank}(k)}{\text{rank}(k)} \tag{2}$$

We illustrate in Fig. 2 how the $\mathcal{H}$-AP is computed for both rankings of Fig. 2b of the main paper. We use the same relevances as in Sec. A.1. The $\mathcal{H}$-AP of the first example is greater (0.78) than of the second one (0.67) because the error is less severe. On the contrary, the AP only considers binary labels and is the same for both rankings (0.45).

One property of AP is that it can be interpreted as the area under the precision-recall curve. $\mathcal{H}$-AP from Eq. (2) can also be interpreted as the area under a hierarchical-precision-recall curve by defining a Hierarchical Recall ($\mathcal{H}$-R@k) and a Hierarchical Precision ($\mathcal{H}$-P@k) as:

$$\mathcal{H}\text{-R@k} = \frac{\sum_{j=1}^{k}\text{rel}(j)}{\sum_{j\in\Omega^+}\text{rel}(j)} \tag{3}$$

$$\mathcal{H}\text{-P@k} = \frac{\sum_{j=1}^{k}\min(\text{rel}(j),\text{rel}(k))}{k\cdot\text{rel}(k)} \tag{4}$$

So that $\mathcal{H}$-AP can be re-written as:

$$\mathcal{H}\text{-AP} = \sum_{k=1}^{|\Omega|}(\mathcal{H}\text{-R@k} - \mathcal{H}\text{-R@k-1}) \times \mathcal{H}\text{-P@k} \tag{5}$$

Eq. (5) recovers Eq. (3) from the main paper, meaning that $\mathcal{H}$-AP generalizes this property of AP beyond binary labels. To further motivate $\mathcal{H}$-AP we will justify the normalization constant for $\mathcal{H}$-AP, and show that $\mathcal{H}$-AP, $\mathcal{H}$-R@k and $\mathcal{H}$-P@k are consistent generalization of AP, R@k, P@k.

Normalization constant for $\mathcal{H}$-AP When all instances are perfectly ranked, all instances j that are ranked before instance k ($s_j \geq s_k$) have a relevance that is higher or equal than k's, *i.e.* $\text{rel}(j) \geq \text{rel}(k)$ and $\min(\text{rel}(j), \text{rel}(k)) = \text{rel}(k)$. So, for each instance k:

$$\begin{aligned}
\mathcal{H}\text{-rank}(k) &= \text{rel}(k) + \sum_{j \in \Omega^+} \min(\text{rel}(k), \text{rel}(j)) \cdot H(s_j - s_k) \\
&= \text{rel}(k) + \sum_{j \in \Omega^+} \text{rel}(k) \cdot H(s_j - s_k) \\
&= \text{rel}(k) \cdot \left(1 + \sum_{j \in \Omega^+} H(s_j - s_k)\right) = \text{rel}(k) \cdot \text{rank}(k)
\end{aligned}$$

The total sum $\sum_{k \in \Omega^+} \frac{\mathcal{H}\text{-rank}(k)}{\text{rank}(k)} = \sum_{k \in \Omega^+} \text{rel}(k)$. This means that we need to normalize by $\sum_{k \in \Omega^+} \text{rel}(k)$ in order to constrain $\mathcal{H}$-AP between 0 and 1. This results in the definition of $\mathcal{H}$-AP from Eq. (2).

$\mathcal{H}$-AP is a consistent generalization of AP In a binary setting, AP is defined as follows:

$$\text{AP} = \frac{1}{|\Omega^+|} \sum_{k \in \Omega^+} \frac{\text{rank}^+(k)}{\text{rank}(k)} \tag{6}$$

$\mathcal{H}$-AP is equivalent to AP in a binary setting ($L = 1$). Indeed, the relevance function is 1 for fine-grained instances and 0 otherwise in the binary case. Therefore $\mathcal{H}\text{-rank}(k) = 1 + \sum_{j \in \Omega^+} H(s_j - s_k)$ which is the same definition as rank^+ in AP. Furthermore the normalization constant of $\mathcal{H}$-AP, $\sum_{k \in \Omega^+} \text{rel}(k)$, is equal to the number of fine-grained instances in the binary setting, *i.e.* $|\Omega^+|$. This means that $\mathcal{H}$-AP = AP in this case.

$\mathcal{H}$-R@k is also a consistent generalization of R@k, indeed:

$$\mathcal{H}\text{-R@k} = \frac{\sum_{j=1}^{k} \text{rel}(j)}{\sum_{j \in \Omega^+} \text{rel}(j)} = \frac{\sum_{j=1}^{k} \mathbb{1}(k \in \Omega^+)}{\sum_{j \in \Omega^+} \mathbb{1}(k \in \Omega^+)} = \frac{\#\text{ number of positive before k}}{|\Omega^+|} = R@k$$

Finally, $\mathcal{H}$-P@k is also a consistent generalization of P@k:

$$\mathcal{H}\text{-P@k} = \frac{\sum_{j=1}^{k} \min(\text{rel}(j), \text{rel}(k))}{k \cdot \text{rel}(k)} = \frac{\#\text{ number of positive before k}}{k} = P@k$$

Link between $\mathcal{H}$-AP and the weighted average of AP Let us define the AP for the semantic level $l \geq 1$ as the binary AP with the set of positives being all instances that belong the same level, *i.e.* $\Omega^{+,l} = \bigcup_{q=l}^{L} \Omega^{(q)}$:

$$\mathrm{AP}^{(l)} = \frac{1}{|\Omega^{+,l}|} \sum_{k \in \Omega^{+,l}} \frac{\mathrm{rank}^{+,l}(k)}{\mathrm{rank}(k)}, \ \mathrm{rank}^{+,l}(k) = 1 + \sum_{j \in \Omega^{+,l}} H(s_j - s_k) \tag{7}$$

Property 1. For any relevance function $\mathrm{rel}(k) = \sum_{p=1}^{l} \frac{w_p}{|\Omega^{+,q}|}$, $k \in \Omega^{(l)}$, with positive weights $\{w_l\}_{l \in [\![1;L]\!]}$ such that $\sum_{l=1}^{L} w_l = 1$:

$$\mathcal{H}\text{-AP} = \sum_{l=1}^{L} w_l \cdot AP^{(l)}$$

i.e. $\mathcal{H}$-AP is equal the weighted average of the AP at all semantic levels.

Proof of Property 1

Denoting $\Sigma w\mathrm{AP} := \sum_{l=1}^{L} w_l \cdot AP^{(l)}$, we obtain from Eq. (7):

$$\Sigma w\mathrm{AP} = \sum_{l=1}^{L} w_l \cdot \frac{1}{|\Omega^{+,l}|} \sum_{k \in \Omega^{+,l}} \frac{\mathrm{rank}^{+,l}(k)}{\mathrm{rank}(k)} \tag{8}$$

We define $\hat{w}_l = \frac{w_l}{|\Omega^{+,l}|}$ to ease notations, so:

$$\Sigma w\mathrm{AP} = \sum_{l=1}^{L} \hat{w}_l \sum_{k \in \Omega^{+,l}} \frac{\mathrm{rank}^{+,l}(k)}{\mathrm{rank}(k)} \tag{9}$$

We define $\mathbb{1}(k,l) = \mathbb{1}\left[k \in \Omega^{+,l}\right]$ so that we can sum over Ω^{+} instead of $\Omega^{+,l}$ and inverse the summations. Note that rank does not depend on l, on contrary to $\mathrm{rank}^{+,l}$.

$$\Sigma w\mathrm{AP} = \sum_{l=1}^{L} \sum_{k \in \Omega^{+}} \frac{\hat{w}_l \cdot \mathbb{1}(k,l) \cdot \mathrm{rank}^{+,l}(k)}{\mathrm{rank}(k)} \tag{10}$$

$$= \sum_{k \in \Omega^{+}} \sum_{l=1}^{L} \frac{\hat{w}_l \cdot \mathbb{1}(k,l) \cdot \mathrm{rank}^{+,l}(k)}{\mathrm{rank}(k)} \tag{11}$$

$$= \sum_{k \in \Omega^{+}} \frac{\sum_{l=1}^{L} \mathbb{1}(k,l) \cdot \hat{w}_l \cdot \mathrm{rank}^{+,l}(k)}{\mathrm{rank}(k)} \tag{12}$$

We replace $\text{rank}^{+,l}$ in Eq. (12) with its definition from Eq. (7):

$$\Sigma w\text{AP} = \sum_{k\in\Omega^+} \frac{\sum_{l=1}^{L} \mathbb{1}(k,l)\cdot\hat{w}_l \cdot \left(1+\sum_{j\in\Omega^{+,l}} H(s_j-s_k)\right)}{\text{rank}(k)} \tag{13}$$

$$= \sum_{k\in\Omega^+} \frac{\sum_{l=1}^{L} \mathbb{1}(k,l)\cdot\hat{w}_l + \sum_{l=1}^{L}\sum_{j\in\Omega^{+,l}} \mathbb{1}(k,l)\cdot\hat{w}_l \cdot H(s_j-s_k)}{\text{rank}(k)} \tag{14}$$

$$= \sum_{k\in\Omega^+} \frac{\sum_{l=1}^{L} \mathbb{1}(k,l)\cdot\hat{w}_l + \sum_{l=1}^{L}\sum_{j\in\Omega^{+}} \mathbb{1}(j,l)\cdot\mathbb{1}(k,l)\cdot\hat{w}_l \cdot H(s_j-s_k)}{\text{rank}(k)} \tag{15}$$

$$= \sum_{k\in\Omega^+} \frac{\sum_{l=1}^{L} \mathbb{1}(k,l)\cdot\hat{w}_l + \sum_{j\in\Omega^{+}}\sum_{l=1}^{L} \mathbb{1}(j,l)\cdot\mathbb{1}(k,l)\cdot\hat{w}_l \cdot H(s_j-s_k)}{\text{rank}(k)} \tag{16}$$

We define the following relevance function:

$$\text{rel}(k) = \sum_{l=1}^{L} \mathbb{1}(k,l)\cdot\hat{w}_l \tag{17}$$

By construction of $\mathbb{1}(\cdot,l)$:

$$\sum_{l=1}^{L} \mathbb{1}(j,l)\cdot\mathbb{1}(k,l)\cdot\hat{w}_l = \min(\text{rel}(k),\text{rel}(j)) \tag{18}$$

Using the definition of the relevance function from Eq. (17) and Eq. (18), we can rewrite Eq. (16) with $\mathcal{H}$-rank:

$$\Sigma w\text{AP} = \sum_{k\in\Omega^+} \frac{\text{rel}(k) + \sum_{j\in\Omega^+}\min(\text{rel}(j),\text{rel}(k))\cdot H(s_j-s_k)}{\text{rank}(k)} \tag{19}$$

$$= \sum_{k\in\Omega^+} \frac{\mathcal{H}\text{-rank}(k)}{\text{rank}(k)} \tag{20}$$

Eq. (20) lacks the normalization constant $\sum_{k\in\Omega^+}\text{rel}(k)$ in order to have the same shape as $\mathcal{H}$-AP in Eq. (2). So we must prove that $\sum_{k\in\Omega^+}\text{rel}(k)=1$:

$$\sum_{k\in\Omega^+}\text{rel}(k) = \sum_{k\in\Omega^+}\sum_{l=1}^{L}\mathbb{1}(k,l)\cdot\hat{w}_l \tag{21}$$

$$= \sum_{l=1}^{L}|\Omega^{(l)}|\sum_{p=1}^{l}\hat{w}_p \tag{22}$$

$$= \sum_{l=1}^{L}|\Omega^{(l)}|\sum_{p=1}^{l}\frac{w_p}{|\Omega^{+,p}|} \tag{23}$$

$$= \sum_{l=1}^{L}|\Omega^{(l)}|\sum_{p=1}^{l}\frac{w_p}{|\bigcup_{q=p}^{L}\Omega^{(q)}|} \tag{24}$$

$$= \sum_{l=1}^{L}|\Omega^{(l)}|\sum_{p=1}^{l}\frac{w_p}{\sum_{q=p}^{L}|\Omega^{(q)}|} \tag{25}$$

$$= \sum_{l=1}^{L}\sum_{p=1}^{l}\frac{|\Omega^{(l)}|\cdot w_p}{\sum_{q=p}^{L}|\Omega^{(q)}|} \tag{26}$$

$$= \sum_{p=1}^{L}\sum_{l=p}^{L}\frac{|\Omega^{(l)}|\cdot w_p}{\sum_{q=p}^{L}|\Omega^{(q)}|} \tag{27}$$

$$= \sum_{p=1}^{L}w_p\cdot\frac{\sum_{l=p}^{L}|\Omega^{(l)}|}{\sum_{q=p}^{L}|\Omega^{(q)}|} \tag{28}$$

$$= \sum_{p=1}^{L}w_p = 1 \tag{29}$$

We have proved that $\Sigma w\text{AP} = \mathcal{H}$-AP with the relevance function of Eq. (17):

$$\Sigma w\text{AP} = \frac{1}{\sum_{k\in\Omega^+}\text{rel}(k)}\sum_{k\in\Omega^+}\frac{\mathcal{H}\text{-rank}(k)}{\text{rank}(k)} = \mathcal{H}\text{-AP} \tag{30}$$

Finally we show, for an instance $k\in\Omega^{(l)}$, :

$$\text{rel}(k) = \sum_{p=1}^{L}\mathbb{1}(k,p)\cdot\hat{w}_p = \sum_{p=1}^{l}\cdot\hat{w}_p = \sum_{p=1}^{l}\frac{w_p}{|\Omega^{+,p}|} \tag{31}$$

i.e. the relevance of Eq. (17) is the same as the relevance of Property 1. This concludes the proof of Property 1. □

A.3 Direct optimisation of $\mathcal{H}$-AP

Decomposing $\mathcal{H}$-rank and rank We have $\Omega^+ = \bigcup_{q=1}^{L} \Omega^{(q)}$, for an instance $k \in \Omega^{(l)}$ we can define the following subsets: $\Omega^{>} = \bigcup_{q=l+1}^{L} \Omega^{(q)}$ and $\Omega^{\leq} = \bigcup_{q=1}^{l} \Omega^{(q)}$, so that $\Omega^+ = \Omega^{>} \cup \Omega^{\leq}$. So we can rewrite $\mathcal{H}$-rank:

$$\begin{aligned}\mathcal{H}\text{-rank}(k) &= \text{rel}(k) + \sum_{j \in \Omega^+} \min(\text{rel}(k), \text{rel}(j)) \cdot H(s_j - s_k) \\ &= \underbrace{\sum_{j \in \Omega^{>}} \min(\text{rel}(k), \text{rel}(j)) \cdot H(s_j - s_k)}_{\mathcal{H}\text{-rank}^{>}} \\ &+ \underbrace{\text{rel}(k) + \sum_{j \in \Omega^{\leq}} \min(\text{rel}(k), \text{rel}(j)) \cdot H(s_j - s_k)}_{\mathcal{H}\text{-rank}^{\leq}}\end{aligned}$$

Similarly we can define $\Omega^{\geq} = \bigcup_{q=l}^{L} \Omega^{(q)}$ and $\Omega^{<} = \bigcup_{q=0}^{l-1} \Omega^{(q)}$, with $\Omega^+ = \Omega^{\geq} \cup \Omega^{<}$. So we can rewrite rank:

$$\begin{aligned}\text{rank}(k) &= 1 + \sum_{k \in \Omega} H(s_j - s_k) \\ &= \underbrace{1 + \sum_{k \in \Omega^{\geq}} H(s_j - s_k)}_{\text{rank}^{\geq}} + \underbrace{\sum_{k \in \Omega^{<}} H(s_j - s_k)}_{\text{rank}^{<}}\end{aligned}$$

Gradients for $\mathcal{L}_{\mathcal{H}\text{-AP}}$ We further decompose $\mathcal{L}_{\mathcal{H}\text{-AP}}$ from Eq. 5 of the main paper, using $\mathcal{H}\text{-rank}^{\leq}(k) = \mathcal{H}\text{-rank}^{=}(k) + \mathcal{H}\text{-rank}^{<}(k)$, $\text{rank}^{\geq}(k) = \text{rank}^{>}(k) + \text{rank}^{=}(k)$:

$$\mathcal{L}_{\mathcal{H}\text{-AP}} = 1 - \frac{1}{\sum_{k \in \Omega^+} \text{rel}(k)} \sum_{k \in \Omega^+} \frac{\mathcal{H}\text{-rank}^{>}(k) + \mathcal{H}\text{-rank}^{=}(k) + \mathcal{H}\text{-rank}^{<}(k)}{\text{rank}^{>}(k) + \text{rank}^{=}(k) + \text{rank}^{<}(k) + \text{rank}^{-}(k)}$$

Table 1: Decomposition of $\mathcal{H}$-AP for optimization.

	$\mathcal{H}\text{-rank}^{>}$	$\text{rank}^{<}$	rank^{-}	$\mathcal{H}\text{-rank}^{=}$	$\mathcal{H}\text{-rank}^{<}$	$\text{rank}^{>}$	$\text{rank}^{=}$
Optimization	✓	✓	✓	✗	✗	✗	✗

We choose to only optimize with respect to the terms indicated with ✓ in Tab. 1.

$\mathbf{rank}^{-}(\boldsymbol{k})$: $\frac{\partial \mathcal{L}_{\mathcal{H}\text{-AP}}}{\partial \operatorname{rank}^{-}(k)} \propto \frac{\mathcal{H}\text{-rank}(k)}{\operatorname{rank}(k)^2} > 0$ which means that in order to decrease $\mathcal{L}_{\mathcal{H}\text{-AP}}$ we must lower rank^{-}, which is an expected behaviour, as it will force k to have a better ranking if it ranked after negative instances (in Ω^{-}).

$\mathbf{rank}^{<}(\boldsymbol{k})$: if we suppose that $\mathcal{H}\text{-rank}^{<}$ is a constant, then $\frac{\partial \mathcal{L}_{\mathcal{H}\text{-AP}}}{\partial \operatorname{rank}^{<}(k)} \propto \frac{\mathcal{H}\text{-rank}(k)}{\operatorname{rank}(k)^2} > 0$ which means that in order to decrease $\mathcal{L}_{\mathcal{H}\text{-AP}}$ we must lower $\operatorname{rank}^{<}$, which is an expected behaviour, as it will force k to have a better ranking if it ranked after negative instances (in $\Omega^{<}$).

$\boldsymbol{\mathcal{H}}\mathbf{\text{-}rank}^{>}(\boldsymbol{k})$: if we suppose that $\operatorname{rank}^{>}$ is a constant, $\frac{\partial \mathcal{L}_{\mathcal{H}\text{-AP}}}{\partial \mathcal{H}\text{-rank}^{>}(k)} \propto \frac{-1}{\operatorname{rank}(k)} < 0$ which means that in order to decrease $\mathcal{L}_{\mathcal{H}\text{-AP}}$ we must increase $\mathcal{H}\text{-rank}^{>}$, which is an expected behaviour, as it will force k to be ranked after other instances of higher relevance (in $\Omega^{>}$).

We choose to not optimize with respect to $\mathcal{H}\text{-rank}^{=}$, $\mathcal{H}\text{-rank}^{<}$, $\operatorname{rank}^{>}$, $\operatorname{rank}^{=}$.

$\mathbf{rank}^{=}$ & $\boldsymbol{\mathcal{H}}\mathbf{\text{-}rank}^{=}$: Optimizing through $\operatorname{rank}^{=}$ has no impact so we choose not to optimize it, indeed $\frac{\partial \mathcal{L}_{\mathcal{H}\text{-AP}}}{\partial \operatorname{rank}^{=}} = 0$. This is the case because inversions between instances of same relevance has no impact on $\mathcal{H}$-AP. This is also the case for $\mathcal{H}\text{-rank}^{=}$.

$\boldsymbol{\mathcal{H}}\mathbf{\text{-}rank}^{<}(\boldsymbol{k})$: $\mathcal{H}\text{-rank}^{<}(k)$ depends on $\operatorname{rank}^{<}(k)$ and the relevance of the other instances that are before. We note that $0 < \frac{\partial \mathcal{H}\text{-rank}^{<}(k)}{\partial \operatorname{rank}(k)} < \operatorname{rel}(k)$ indeed when the $\operatorname{rank}^{<}$ increases $\mathcal{H}\text{-rank}^{<}$ increases and the increase rate can not be equal or greater than $\operatorname{rel}(k)$

$$\frac{\partial \mathcal{L}_{\mathcal{H}\text{-AP}}}{\partial \operatorname{rank}^{<}(k)} \propto -\Bigg(\overbrace{\left(\frac{\partial \mathcal{H}\text{-rank}^{<}(k)}{\partial \operatorname{rank}^{<}(k)} - \operatorname{rel}(k) \right) \cdot \operatorname{rank}^{>}(k)}^{a} \tag{32}$$

$$+ \overbrace{\left(\frac{\partial \mathcal{H}\text{-rank}^{<}(k)}{\partial \operatorname{rank}^{<}(k)} - \operatorname{rel}(k) \right) \cdot \operatorname{rank}^{=}(k)}^{b} \tag{33}$$

$$+ \overbrace{\left(\frac{\partial \mathcal{H}\text{-rank}^{<}(k)}{\partial \operatorname{rank}^{<}(k)} \cdot \operatorname{rank}^{<}(k) - \mathcal{H}\text{-rank}^{<}(k) \right)}^{c} \tag{34}$$

$$+ \overbrace{\frac{\partial \mathcal{H}\text{-rank}^{<}(k)}{\partial \operatorname{rank}^{<}(k)} \cdot \operatorname{rank}^{-}(k)}^{d} \Bigg) / \operatorname{rank}(k)^2 \tag{35}$$

When optimizing through $\mathcal{H}\text{-rank}^{<}$ we can no longer explicitly control the sign of $\frac{\partial \mathcal{L}_{\mathcal{H}\text{-AP}}}{\partial \operatorname{rank}^{<}(k)}$. For example if a and b are null (*i.e.* not instances of higher or equal relevance are above k), d remains and is greater than 0 and c can be greater than 0 resulting in an overall negative gradient, which is an unexpected behaviour. This is why we choose to not optimize through $\mathcal{H}\text{-rank}^{<}$.

rank$^>(k)$: We have $\mathcal{H}$-rank$^>(k) = \text{rel}(k) \cdot \text{rank}^>(k)$ indeed all instances j ranked before k have a strictly higher relevance, *i.e.* $\min(\text{rel}(j), \text{rel}(k)) = \text{rel}(k)$, so we can write:

$$\frac{\partial \mathcal{L}_{\mathcal{H}\text{-AP}}}{\partial\, \text{rank}^>(k)} \propto \frac{\overbrace{\mathcal{H}\text{-rank}^<(k) - \text{rel}(k) \cdot \text{rank}^<(k)}^{<0} - \text{rel}(k) \cdot \text{rank}^-(k)}{\text{rank}(k)^2} < 0 \tag{36}$$

Optimizing trough rank$^>$ instead of only $\mathcal{H}$-rank$^>$ diminishes the magnitude of the resulting gradient, so we decide to not optimize through rank$^>$.

Approximating $\mathcal{H}$-rank$^>$ In order to have a lower bound on $\mathcal{H}$-rank$^>$ we approximate the Heaviside step function H with a smooth lower bound:

$$H_s^>(t) = \begin{cases} \gamma \cdot t, & \text{if } t < 0 \\ \max(\nu \cdot t + \mu, 1), & \text{if } t \geq 0 \end{cases} \tag{37}$$

$H_s^>$ is illustrated in Fig. 3a. Using $H_s^>$ we can approximate $\mathcal{H}$-rank$^>$: $\mathcal{H}$-rank$_s^>(k) = \text{rel}(k) + \sum_{j \in \Omega^+} \min(\text{rel}(j), \text{rel}(k)) H_s^>(s_j - s_k)$. Because $H_s^>(t) \leq H(t)$: $\mathcal{H}$-rank$_s^>(k) \leq \mathcal{H}$-rank$^>$. In our experiments we use: $\gamma = 10$, $\nu = 25$, $\mu = 0.5$.

Approximating rank$^<$ In order to have an upper bound on rank$^<$ we approximate the Heaviside with a smooth upper bound as given in [21]:

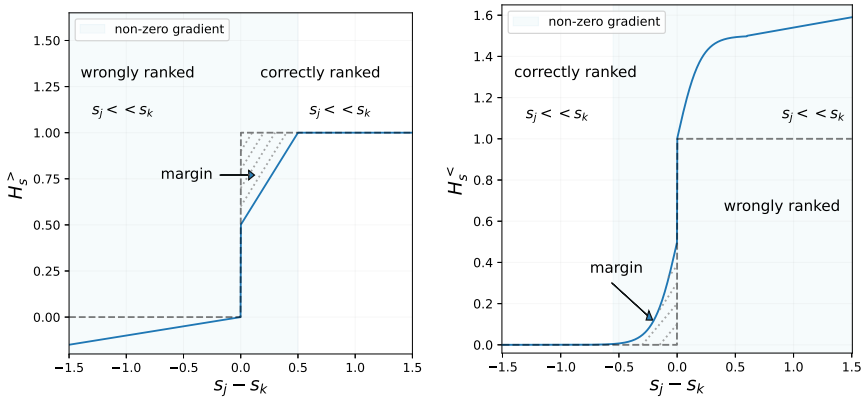
$$H_s^<(t) = \begin{cases} \sigma(\frac{t}{\tau}) & \text{if } t \leq 0, \quad \text{where } \sigma \text{ is the sigmoid function} \\ \sigma(\frac{t}{\tau}) + 0.5 & \text{if } t \in [0; \delta] \quad \text{with } \delta \geq 0 \\ \rho \cdot (t - \delta) + \sigma(\frac{\delta}{\tau}) + 0.5 & \text{if } t > \delta \end{cases} \tag{38}$$

$H_s^<$ is illustrated in Fig. 3a. Using $H_s^<$ we can approximate rank$^<$: rank$_s^<(k) = 1 + \sum_{j \in \Omega} H_s^<(s_j - s_k)$. Because $H_s^<(t) \geq H(t)$: rank$_s^<(k) \geq$ rank$^<$. We use the hyper-parameters: $\tau = 0.01$, $\rho = 100$, $\delta = 0.05$.

We illustrate in Fig. 3a $H_s^>$ and in Fig. 3a $H_s^<$ *vs.* $s_j - s_k$. The margins denote the fact the even when the instance k is correctly ranked (lower cosine similarity than j in Fig. 3a and higher in Fig. 3a) we still want to backbropagate gradient which leads to more robust training.

A.4 Discussion

HAPPIER requires the definition of the relevance function. In our work, we leverage the hierarchical tree between concepts to this end. Is this a strong assumption? We argue that the access to a hierarchical tree is not a prohibitive factor. Hierarchical trees are available for a surprising number of datasets (CUB-200-2011 [25], Cars196 [12], InShop [13], SOP [17]), including *large scale* ones



(a) Illustration of $H_s^>$ in Eq. (37). (b) Illustration of $H_s^<$ in Eq. (37).

Fig. 3: Illustrations of the two approximations of the Heaviside step function used to approximate $\mathcal{H}$-rank$^>$ and rank$^<$.

(iNaturalist [24], the three DyML datasets [22] and also Imagenet [6]). Even when hierarchical relations are not directly available, they are not that difficult to obtain since the tree complexity depends only on the number of classes and not of examples. Hierarchical relations can be semi-automatically obtained by grouping fine-grained labels in existing datasets, as was previously done by *e.g.* [4]. For instance, while hierarchical labels are not directly available in scene or landmarks datasets [20], this could be extended to them at a reasonable cost, *e.g.* in Paris6k "Sacre Coeur" might be considered closer to "Notre Dame" than to the "Moulin Rouge". The large lexical database Wordnet [15] can also be used to define hierarchies between labels and define semantic similarity, as in Imagenet [6] or the SUN database [26]. Furthermore, our approach can be extended to leverage general knowledge beyond hierarchical taxonomies, by defining more general relevance functions built on *e.g.* continuous similarities or attributes [18].

B Experiments

B.1 Datasets

Stanford Online Product (SOP) [17] is a standard dataset for Image Retrieval it has two levels of semantic scales, the object Id (fine) and the object category (coarse). It depicts Ebay online objects, with 120 053 images of 22 634 objects (Id) classified into 12 (coarse) categories (*e.g.* bikes, coffee makers *etc.*), see Fig. 4. We use the reference train and test splits from [17]. The dataset can be downloaded at: https://cvgl.stanford.edu/projects/lifted_struct/.

iNaturalist-2018 Base/Full iNaturalist-2018 is a dataset that has been used for Image Retrieval in recent works [1,21]. It depicts animals, plants, mushroom

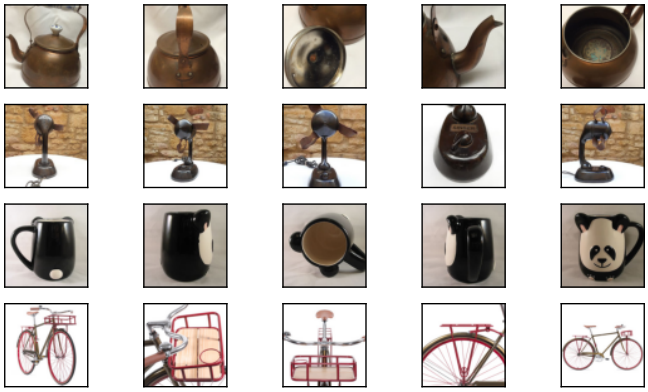


Fig. 4: Images from Stanford Online Products.

etc. in wildlife, see Fig. 5, it has in total 461 939 images and 8142 fine-grained classes ("Species"). We use two different sets of annotations: a set of annotations with 2 semantic levels the species (fine) and intermediate scale (coarse), we term this dataset iNat-base, and the full biological taxonomy which consists of 7 semantic levels ("Species", "Genus" ...) we term this dataset iNat-full. We use the standard Image Retrieval splits from [1]. The dataset can be downloaded at: `github.com/visipedia/inat_comp`, and the retrieval splits at: `drive.google.com`.

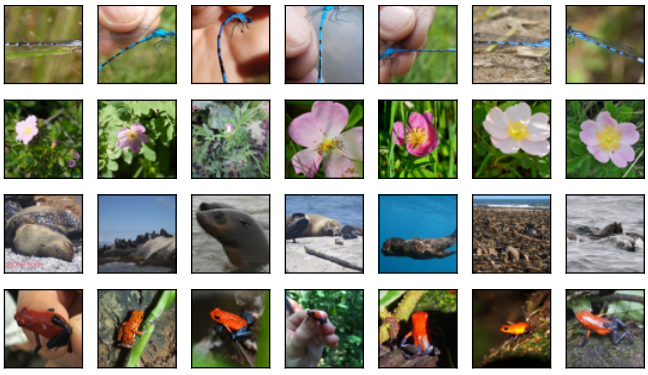


Fig. 5: Images from iNaturalist-2018.

DyML-datasets The DyML benchmark [22] is composed of three datasets, DyML-V that depicts vehicles, DyML-A that depicts animals, DyML-P that depicts online products. The training set has three levels of semantic ($L = 3$),

and each image is annotated with the label corresponding to each level (like SOP and iNat-base/full), however the test protocol is different. At test time for each dataset there is three sub-datasets, each sub-dataset aims at evaluating the model on a specific hierarchical level (*e.g.* "Fine"), so we can only compute binary metrics on each sub-dataset. We describe in Tab. 2 the statistics of the train and test datasets. The three datasets can be downloaded at: `onedrive.live.com`.

Table 2: Statistics of the three train and test DyML benchmarks [22].

Datasets		DyML-Vehicle		DyML-Animal		DyML-Product	
		train	test	train	test	train	test
Coarse	Classes	5	6	5	5	36	6
	Images	343.1 K	5.9 K	407.8 K	12.5 K	747.1 K	1.5 K
Middle	Classes	89	127	28	17	169	37
	Images	343.1 K	34.3 K	407.8 K	23.1 K	747.1 K	1.5 K
Fine	Classes	36,301	8,183	495	162	1,609	315
	Images	343.1 K	63.5 K	407.8 K	11.3 K	747.1 K	1.5 K

B.2 Implementation details

SOP & iNat-base/full Our model is a ResNet-50 [9] pretrained on Imagenet to which we append a `LayerNormalization` layer with no affine parameters after the (average) pooling and a Linear layer that reduces the embeddings size from 2048 to 512. We use the Adam [11] optimizer with a base learning rate of $1e^{-5}$ and weight decay of $1e^{-4}$ for SOP and a base learning rate of $1e^{-5}$ and weight decay of $4e^{-4}$ for iNat-base/full. The learning rate is decreased using cosine annealing decay, for 75 epochs on SOP and 100 epochs on iNat-base/full. We "warm up" our model for 5 epochs, *i.e.* the pretrained weights are not optimized. We use standard data augmentation: `RandomResizedCrop` and `RandomHorizontalFlip`, with a final crop size of 224, at test time we use `CenterCrop`. We set the random seed to 0 in all our experiments. We use a fixed batch size of 256 and use the hard sampling strategy from [2] on SOP and the standard class balanced sampling [29] (4 instances per class) on iNat-base/full.

DyML We use a ResNet-34 [9] randomly initialized on DyML-V&A and pretrained on Imagenet for DyML-P, following [22]. We use an SGD optimizer with Nesterov momentum (0.9), a base learning rate of 0.1 on DyML-V&A and 0.01 on DyML-P with a weight decay of $1e^{-4}$. We use cosine annealing decay to reduce the learning rate for 100 epochs on DyML-V&A and 20 on DyML-P. We use the same data augmentation and random seed as for SOP and iNat-base. We also use the class balanced sampling (4 instances per class) with a fixed batch size of 256.

B.3 Metrics

The ASI [7] measures at each rank $n \leq N$ the set intersection proportion (SI) between the ranked list $a_1, \ldots, a_N$ and the ground truth ranking $b_1, \ldots, b_N$, with N the total number of positives. As it compares intersection the ASI can naturally take into account the different levels of semantic:

$$SI(n) = \frac{|\{a_1, \ldots, a_n\} \cap \{b_1, \ldots, b_n\}|}{n}$$
$$ASI = \frac{1}{N} \sum_{n=1}^{N} SI(n)$$

The NDCG [5] is the reference metric in information retrieval, we define it using the semantic level l of each instance:

$$DCG = \sum_{k \in \Omega^+} \frac{2^l - 1}{\log_2(1 + \text{rank}(k))}, \text{ with } k \in \Omega^{(l)}.$$
$$NDCG = \frac{DCG}{\max_{\text{ranking}} DCG}$$

To compute the AP for the semantic level l we consider that all instances with semantic levels $\geq l$ are positives:

$$AP^{(l)} = \sum_{k \in \bigcup_{q=l}^{L} \Omega^{(q)}} \frac{\text{rank}^l(k)}{\text{rank}(k)}, \text{ where } \text{rank}^l(k) = 1 + \sum_{j \in \bigcup_{q=l}^{L} \Omega^{(q)}} H(s_j - s_k)$$

B.4 Source Code

Our code is based on `PyTorch` [19]. We use utilities from `Pytorch Metric Learning` [16] *e.g.* for samplers and losses, `Hydra` [28] to handle configuration files (Yaml), `tsnecuda` [3] to compute t-SNE reductions using GPUs and standard Python libraries such as `NumPy` [8] or `Matplotlib` [10].

We use the publicly available implementations of the NSM loss [29][1] which is under an Apache-2.0 license, of NCA++[23][2] which is under an MIT license, of ROADMAP [21][3] which is under an MIT license, we use the implementation of `Pytorch Metric Learning` [16][4] for the TL_{SH} [27] (MIT license), and finally we have implemented the CSL [22] after discussion with the authors and we will make it part of our repository.

[1] https://github.com/azgo14/classification_metric_learning
[2] https://github.com/euwern/proxynca_pp
[3] https://github.com/elias-ramzi/ROADMAP
[4] https://github.com/KevinMusgrave/pytorch-metric-learning

We had access to both Nvidia Quadro RTX 5000 and Tesla V-100 (16 GiB GPUs). We use mixed precision training [14], which is native to `PyTorch`, to accelerate training, making our models train for up to 7 hours on Stanford Online Products, 25 hours on iNaturalist-2018, less than 20 hours on both DyML-A and DyML-V and 6 hours on DyML-P.

B.5 On DyML results

Their is no public code available to reproduce the results of [22]. After personal correspondence with the authors, we have been able to re-implement the CSL method from [22]. We report the differences in performances between our results and theirs in Tab. 3. Our implementation of CSL performs better on the three datasets which is the results of our better training recipes detailed in Sec. B.2. Our discussions with the authors of [22] confirmed that the performances obtained with our re-implementation of CSL are valid and representative of the method's potential.

Table 3: Difference in performances for CSL between results reported in [22] and our experiments on the DyML benchmarks.

Method	DyML-Vehicle			DyML-Animal			DyML-Product		
	mAP	ASI	R@1	mAP	ASI	R@1	mAP	ASI	R@1
CSL [22]	12.1	23.0	25.2	31.0	45.2	52.3	28.7	29.0	54.3
CSL (ours)	30.0	43.6	87.1	40.8	46.3	60.9	31.1	40.7	52.7

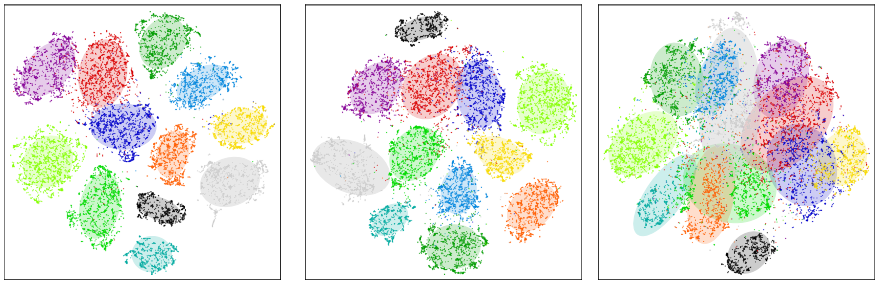
C Qualitative results

C.1 Robustness to λ

Fig. 4b of the main paper illustrates that HAPPIER is robust with respect to λ with performances increasing for most values between 0.1 and 0.9. In addition, we also show in Fig. 6 that for $0 < \lambda < 0.9$ HAPPIER leads to a better organization of the embedding space than a fine-grained baseline (see Fig. 4a in main paper). This is expected since the lower λ is, the more emphasis is put on optimizing $\mathcal{H}$-AP, which organizes the embedding space in a hierarchical structure.

C.2 Comparison of HAPPIER *vs.* CSL

In Fig. 7, we compare HAPPIER against the hierarchical image retrieval method CSL [22]. We observe qualitative improvements where HAPPIER results in a better ranking. This highlights the benefit of optimizing directly a hierarchical metric, *i.e.* $\mathcal{H}$-AP, rather than optimizing a proxy based triplet loss as in CSL.



(a) $\lambda = 0.3$ (b) $\lambda = 0.5$ (c) $\lambda = 0.9$

Fig. 6: t-SNE visualisation of the embedding space of models trained with HAPPIER on SOP with different values of λ. Each point is the average embedding of each fine-grained label (object instance) and the colors represent coarse labels (object category, *e.g.* bike, coffee maker).

C.3 Controlled errors: iNat-base

We showcase in Fig. 8 errors of HAPPIER *vs.* a fine-grained baseline on iNat-base. On Fig. 8a, we illustrate how a model trained with HAPPIER makes mistakes that are less severe than a baseline model trained only on the fine-grained level. On Fig. 8b, we show an example where both models fail to retrieve the correct fine-grained instances, however the model trained with HAPPIER retrieves images of bikes that are semantically more similar to the query.

C.4 Controlled errors: iNat-full

We illustrate in Figs. 9 and 10 an example of a query image and the top 25 retrieved results on iNat-full ($L = 7$). Given the same query both models failed to retrieve the correct fine-grained images (that would be in $\Omega^{(7)}$). The standard model in Fig. 10 retrieves images that are semantically more distant than the images retrieved with HAPPIER in Fig. 9. For example HAPPIER retrieves images that are either in $\Omega^{(5)}$ or $\Omega^{(4)}$ (only one instance is in $\Omega^{(3)}$) whereas the standard model retrieves instances that are in $\Omega^{(2)}$ or $\Omega^{(1)}$.

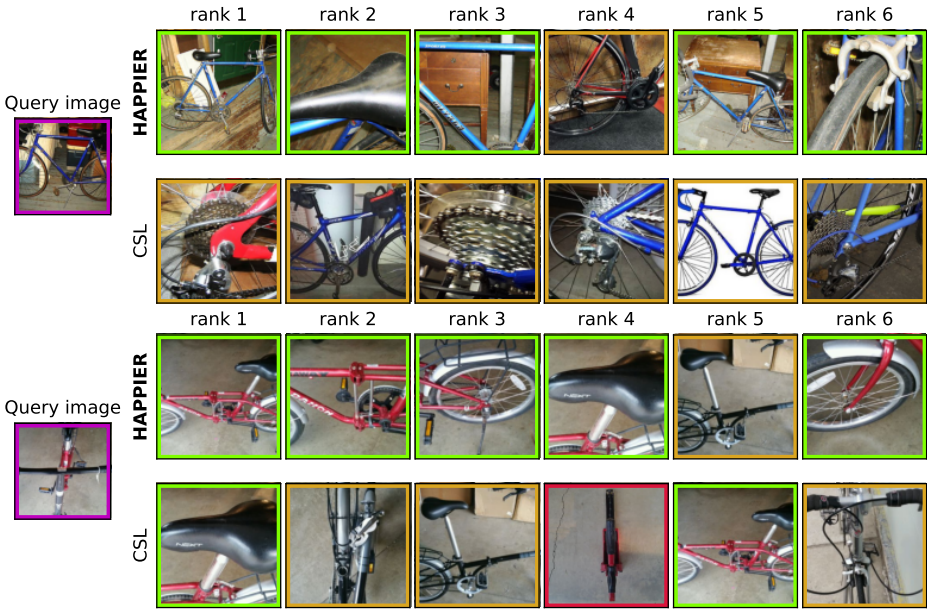
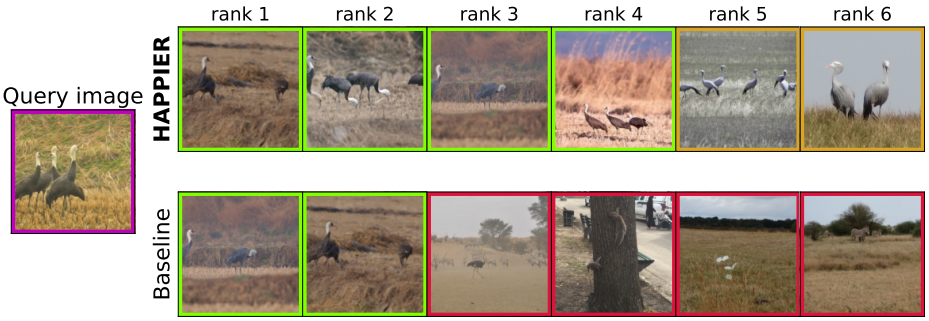
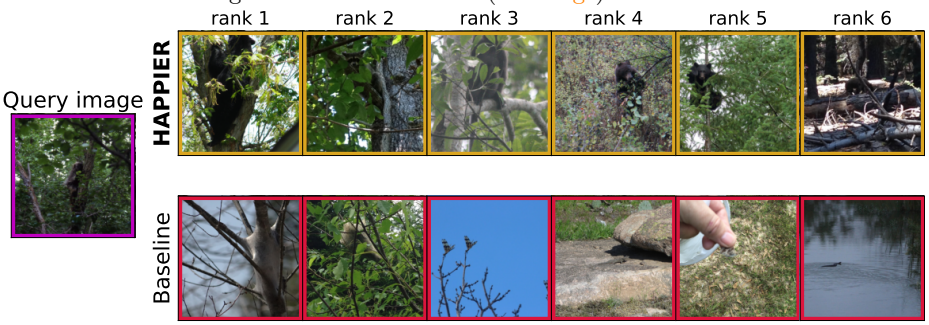


Fig. 7: Qualitative comparison of HAPPIER *vs.* CSL [22] on SOP



(a) HAPPIER can help make less severe mistakes. The inversion on the bottom row are with negative instances (in red), where as with HAPPIER (top row) inversions are with instances sharing the same coarse label (in orange).



(b) In this example, the models fail to retrieve the correct fine grained images. However HAPPIER still retrieves images with the same coarse label (in orange) whereas the baseline retrieves images that are dissimilar semantically to the query (in red).

Fig. 8: Qualitative examples of failure cases from a standard fine-grained model corrected by training with HAPPIER.

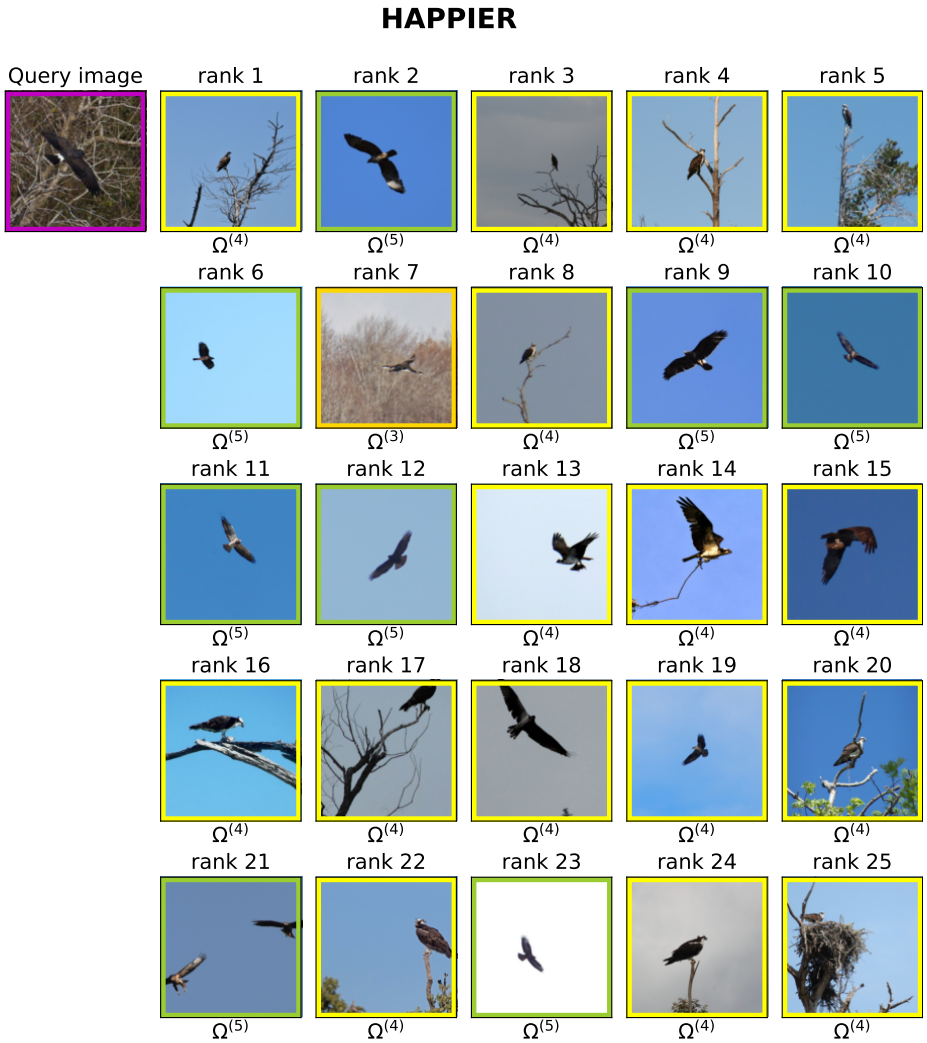


Fig. 9: Images retrieved for the query image by a model trained with **HAPPIER** on iNat-full ($L = 7$).

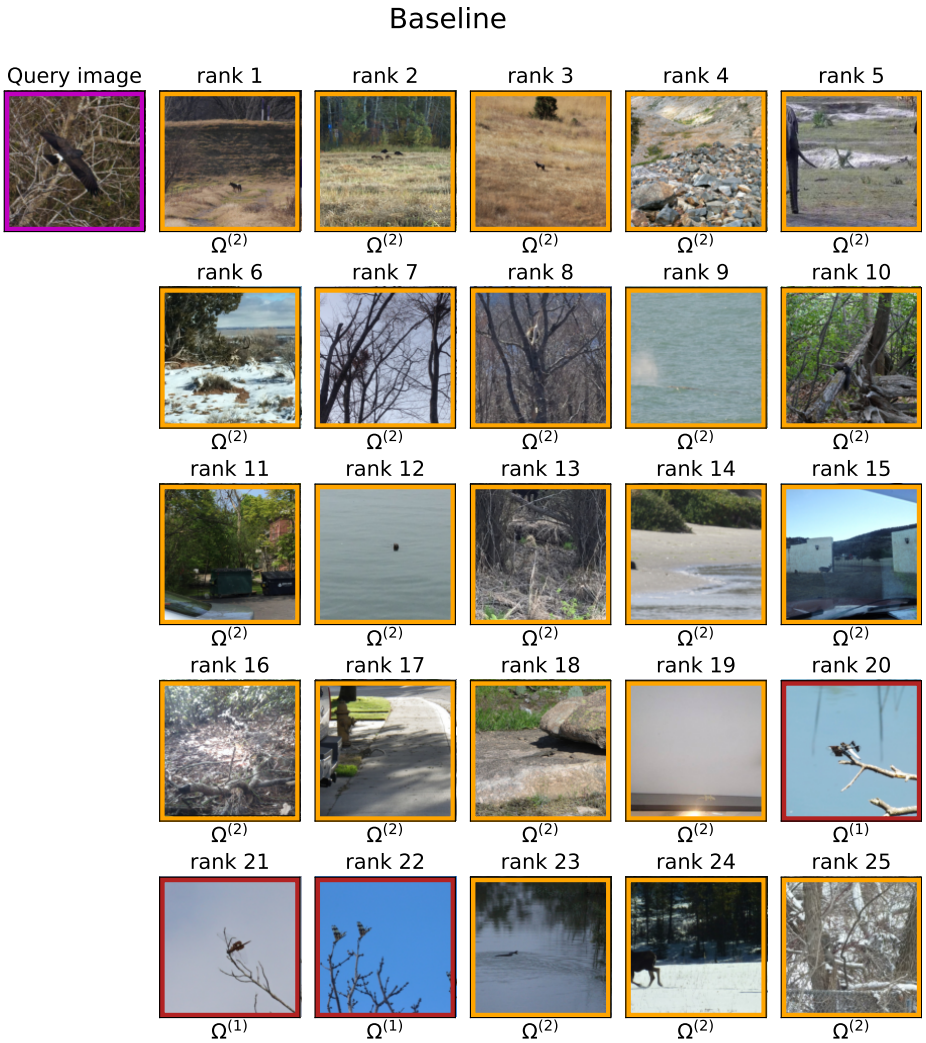


Fig. 10: Images retrieved for the query image by a model trained with standard model on iNat-full ($L = 7$).

References

1. Brown, A., Xie, W., Kalogeiton, V., Zisserman, A.: Smooth-ap: Smoothing the path towards large-scale image retrieval. In: European Conference on Computer Vision. pp. 677–694. Springer (2020) 10, 11
2. Cakir, F., He, K., Xia, X., Kulis, B., Sclaroff, S.: Deep metric learning to rank. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 1861–1870 (2019) 12
3. Chan, D.M., Rao, R., Huang, F., Canny, J.F.: Gpu accelerated t-distributed stochastic neighbor embedding. Journal of Parallel and Distributed Computing **131**, 1–13 (2019) 13
4. Chang, D., Pang, K., Zheng, Y., Ma, Z., Song, Y.Z., Guo, J.: Your" flamingo" is my" bird": Fine-grained, or not. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 11476–11485 (2021) 10
5. Croft, W.B., Metzler, D., Strohman, T.: Search engines: Information retrieval in practice, vol. 520. Addison-Wesley Reading (2010) 13
6. Deng, J., Dong, W., Socher, R., Li, L.J., Li, K., Fei-Fei, L.: Imagenet: A large-scale hierarchical image database. In: 2009 IEEE Conference on Computer Vision and Pattern Recognition. pp. 248–255 (2009). https://doi.org/10.1109/CVPR.2009.5206848 10
7. Fagin, R., Kumar, R., Sivakumar, D.: Comparing top k lists. SIAM Journal on discrete mathematics **17**(1), 134–160 (2003) 13
8. Harris, C.R., Millman, K.J., van der Walt, S.J., Gommers, R., Virtanen, P., Cournapeau, D., Wieser, E., Taylor, J., Berg, S., Smith, N.J., Kern, R., Picus, M., Hoyer, S., van Kerkwijk, M.H., Brett, M., Haldane, A., del R'ıo, J.F., Wiebe, M., Peterson, P., G'erard-Marchant, P., Sheppard, K., Reddy, T., Weckesser, W., Abbasi, H., Gohlke, C., Oliphant, T.E.: Array programming with NumPy. Nature **585**(7825), 357–362 (Sep 2020). https://doi.org/10.1038/s41586-020-2649-2, https://doi.org/10.1038/s41586-020-2649-2 13
9. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. corr abs/1512.03385 (2015) (2015) 12
10. Hunter, J.D.: Matplotlib: A 2d graphics environment. Computing in Science & Engineering **9**(3), 90–95 (2007). https://doi.org/10.1109/MCSE.2007.55 13
11. Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980 (2014) 12
12. Krause, J., Stark, M., Deng, J., Fei-Fei, L.: 3d object representations for fine-grained categorization. In: 4th International IEEE Workshop on 3D Representation and Recognition (3dRR-13). Sydney, Australia (2013) 9
13. Liu, Z., Luo, P., Qiu, S., Wang, X., Tang, X.: Deepfashion: Powering robust clothes recognition and retrieval with rich annotations. In: Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (June 2016) 9
14. Micikevicius, P., Narang, S., Alben, J., Diamos, G., Elsen, E., Garcia, D., Ginsburg, B., Houston, M., Kuchaiev, O., Venkatesh, G., et al.: Mixed precision training. arXiv preprint arXiv:1710.03740 (2017) 14
15. Miller, G.A.: Wordnet: A lexical database for english. Commun. ACM **38**(11), 39–41 (nov 1995). https://doi.org/10.1145/219717.219748, https://doi.org/10.1145/219717.219748 10
16. Musgrave, K., Belongie, S., Lim, S.N.: Pytorch metric learning (2020) 13
17. Oh Song, H., Xiang, Y., Jegelka, S., Savarese, S.: Deep metric learning via lifted structured feature embedding. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 4004–4012 (2016) 9, 10

18. Parikh, D., Grauman, K.: Relative attributes. In: 2011 International Conference on Computer Vision. pp. 503–510. IEEE (2011) 10
19. Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., Desmaison, A., Kopf, A., Yang, E., DeVito, Z., Raison, M., Tejani, A., Chilamkurthy, S., Steiner, B., Fang, L., Bai, J., Chintala, S.: Pytorch: An imperative style, high-performance deep learning library. In: Wallach, H., Larochelle, H., Beygelzimer, A., d'Alché-Buc, F., Fox, E., Garnett, R. (eds.) Advances in Neural Information Processing Systems 32, pp. 8024–8035. Curran Associates, Inc. (2019) 13
20. Radenović, F., Iscen, A., Tolias, G., Avrithis, Y., Chum, O.: Revisiting oxford and paris: Large-scale image retrieval benchmarking. In: CVPR (2018) 10
21. Ramzi, E., Thome, N., Rambour, C., Audebert, N., Bitot, X.: Robust and decomposable average precision for image retrieval. Advances in Neural Information Processing Systems **34** (2021) 9, 10, 13
22. Sun, Y., Zhu, Y., Zhang, Y., Zheng, P., Qiu, X., Zhang, C., Wei, Y.: Dynamic metric learning: Towards a scalable metric space to accommodate multiple semantic scales. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 5393–5402 (2021) 10, 11, 12, 13, 14, 16
23. Teh, E.W., DeVries, T., Taylor, G.W.: Proxynca++: Revisiting and revitalizing proxy neighborhood component analysis. In: European Conference on Computer Vision. pp. 448–464. Springer (2020) 13
24. Van Horn, G., Mac Aodha, O., Song, Y., Cui, Y., Sun, C., Shepard, A., Adam, H., Perona, P., Belongie, S.: The inaturalist species classification and detection dataset. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 8769–8778 (2018) 10
25. Wah, C., Branson, S., Welinder, P., Perona, P., Belongie, S.: The caltech-ucsd birds-200-2011 dataset. Tech. Rep. CNS-TR-2011-001, California Institute of Technology (2011) 9
26. Wilt, E.W., Harrison, A.V.: Creating a semantic hierarchy of SUN database object labels using WordNet. In: Pham, T., Solomon, L. (eds.) Artificial Intelligence and Machine Learning for Multi-Domain Operations Applications III. vol. 11746, pp. 76 – 83. International Society for Optics and Photonics, SPIE (2021). https://doi.org/10.1117/12.2588827, https://doi.org/10.1117/12.2588827 10
27. Wu, C.Y., Manmatha, R., Smola, A.J., Krahenbuhl, P.: Sampling matters in deep embedding learning. In: Proceedings of the IEEE International Conference on Computer Vision. pp. 2840–2848 (2017) 13
28. Yadan, O.: Hydra - a framework for elegantly configuring complex applications. Github (2019), https://github.com/facebookresearch/hydra 13
29. Zhai, A., Wu, H.Y.: Classification is a strong baseline for deep metric learning. arXiv preprint arXiv:1811.12649 (2018) 12, 13