Deep Hashing Network for Unsupervised Domain Adaptation
Supplementary Material

Hemanth Venkateswara, Jose Eusebio, Shayok Chakraborty, Sethuraman Panchanathan
Center for Cognitive Ubiquitous Computing, Arizona State University, Tempe, AZ, USA
{hemanthv, jeusebio, shayok.chakraborty, panch}@asu.edu

1. Loss Function Derivative

In this section we outline the derivative of Equation 8 for the backpropagation algorithm;

$$\min_{\mathcal{U}} J = \mathcal{L}(u_s) + \gamma \mathcal{M}(u_s, u_t) + \eta \mathcal{H}(u_s, u_t),$$

(8)

where, $\mathcal{U} := \{u_s \cup u_t\}$ and $(\gamma, \eta)$ control the importance of domain adaptation (1) and target entropy loss (7) respectively. In the following subsections, we outline the derivative of the individual terms w.r.t. the input $u$. 

1.1. Derivative for MK-MMD

$$\mathcal{M}(u_s, u_t) = \sum_{i \in \mathcal{X}} d_k^2(u_s^i, u_t^i),$$

(1)

$$d_k^2(u_s^i, u_t^i) = \left|\left| \mathbb{E}[\phi(u_s^{i,l})] - \mathbb{E}[\phi(u_t^{i,l})] \right|\right|^2_{\mathcal{H}_k}.$$ 

(2)

We implement the linear MK-MMD loss according to [1]. For this derivation, we consider the loss at just one layer. The derivative for the MK-MMD loss at every other layer can be derived in a similar manner. The output of $i^{th}$ source data point at layer $l$ is represented as $u_s$, and the output of the $i^{th}$ target data point is represented as $v_t$. For ease of representation, we drop the superscripts for the source ($s$), the target ($t$) and the layer ($l$). Unlike the conventional MMD loss which is $O(n^2)$, the MK-MMD loss outlined in [1] is $O(n)$ and can be estimated online (does not require all the data). The loss is calculated over every batch of data points during the back-propagation. Let $n$ be the number of source data points $\mathcal{U} := \{u_s\}^{n}_{i=1}$ and the number of target data points $\mathcal{V} := \{v_t\}^{n}_{i=1}$ in the batch. We assume equal number of source and target data points in a batch and that $n$ is even. The MK-MMD is defined over a set of 4 data points $w_i = [u_{2i-1}, u_{2i}, v_{2i-1}, v_{2i}], \forall i \in \{1, 2, \ldots, n/2\}$. The MK-MMD is given by,

$$\mathcal{M}(u, v) = \sum_{m=1}^{\kappa} \beta_m \frac{1}{n/2} \sum_{i=1}^{n/2} h_m(w_i),$$

(9)

where, $\kappa$ is the number of kernels and $\beta_m = 1/\kappa$ is the weight for each kernel and,

$$h_m(w_i) = k_m(u_{2i-1}, u_{2i}) + k_m(v_{2i-1}, v_{2i}) - k_m(u_{2i-1}, v_{2i}) - k_m(u_{2i}, v_{2i-1}),$$

(10)

where, $k_m(x, y) = \exp(-\frac{||x-y||^2_2}{\sigma^2_m})$. Re-writing the MK-MMD in terms of the kernels, we have,

$$\mathcal{M}(u, v) = \frac{2}{n\kappa} \sum_{m=1}^{\kappa} \sum_{i=1}^{n/2} \left[ k_m(u_{2i-1}, u_{2i}) + k_m(v_{2i-1}, v_{2i}) - k_m(u_{2i-1}, v_{2i}) - k_m(u_{2i}, v_{2i-1}) \right],$$

(11)
We now outline the derivative of 11 w.r.t. source output $u_q$ and target output $v_q$. The derivative is,

$$
\frac{\partial M}{\partial u_q} = \frac{2}{nk} \sum_{m=1}^{n/2} \sum_{i=1}^{n/2} \left[ \frac{2}{\sigma_m} k_m(u_{2i-1}, u_{2i}).(u_{2i-1} - u_{2i}).(I\{q = 2i\} - I\{q = 2i - 1\}) \right. \\
+ \left. \frac{2}{\sigma_m} k_m(u_{2i-1}, v_{2i}).(v_{2i-1} - v_{2i}).I\{q = 2i\} - I\{q = 2i - 1\}) \right], 
$$

(12)

where, $I\{\cdot\}$ is the indicator function which is 1 if the condition is true, else it is false. The derivative w.r.t. the target data output $v_q$ is,

$$
\frac{\partial M}{\partial v_q} = \frac{2}{nk} \sum_{m=1}^{n/2} \sum_{i=1}^{n/2} \left[ \frac{2}{\sigma_m} k_m(v_{2i-1}, v_{2i}).(v_{2i-1} - v_{2i}).(I\{q = 2i\} - I\{q = 2i - 1\}) \right. \\
- \left. \frac{2}{\sigma_m} k_m(u_{2i-1}, v_{2i}).(u_{2i-1} - v_{2i}).I\{q = 2i\} - I\{q = 2i - 1\}) \right], 
$$

(13)

1.2. Derivative for Supervised Hash Loss

The supervised hash loss is given by,

$$
\min_{u_s} L(u_s) = -\sum_{s_{ij} \in S} \left( s_{ij} u_i^\top u_j - \log(1 + \exp(u_i^\top u_j)) \right) \\
+ \frac{n_s}{2} \sum_{i=1}^{n_s} \|u_i - \text{sgn}(u_i)\|^2_2.
$$

(5)

The partial derivative of 5 w.r.t. source data output $u_p$ is given by,

$$
\frac{\partial L}{\partial u_p} = \sum_{s_{ij} \in S} \left[ I\{i = q\} \left( \sigma(u_i^\top u_j) - s_{ij} \right) u_j + I\{j = q\} \left( \sigma(u_i^\top u_j) - s_{ij} \right) u_i \right] + 2(u_q - \text{sgn}(u_q)),
$$

(14)

where, $\sigma(x) = \frac{1}{1 + \exp(-x)}$. We assume $\text{sgn}(\cdot)$ to be a constant and avoid the differentiability issues with $\text{sgn}(\cdot)$ at 0. Since the $S$ is symmetric, we can reduce the derivative to,

$$
\frac{\partial L}{\partial u_q} = \sum_{j=1}^{n_s} \left[ 2(\sigma(u_q^\top u_j) - s_{qj}) u_j \right] + 2(u_q - \text{sgn}(u_q)).
$$

(15)

1.3. Derivative for Unsupervised Entropy Loss

We outline the derivative of $\frac{\partial H}{\partial t}$ in the following section, where $H$ is defined as,

$$
H(u_s, u_t) = -\frac{1}{n_t} \sum_{i=1}^{n_t} \sum_{j=1}^{C} p_{ij} \log(p_{ij})
$$

(7)

and $p_{ij}$ is the probability of target data output $u^t_i$ belonging to category $j$, given by

$$
p_{ij} = \frac{\sum_{k=1}^{K} \exp(u_{ij}^t u_k^s)}{\sum_{k=1}^{K} \sum_{l=1}^{C} \exp(u_{ij}^t u_k^s) / u_{ij}^t}
$$

(6)

For ease of representation, we will denote the target output $u^t_i$ as $v_i$ and drop the superscript $t$. Similarly, we will denote the $k^{th}$ source data point in the $j^{th}$ category $u_k^s$ as $u_k^s$, by dropping the domain superscript. We define the probability $p_{ij}$ with the news terms as,

$$
p_{ij} = \frac{\sum_{k=1}^{K} \exp(v_i^\top u_k^s)}{\sum_{l=1}^{C} \sum_{k'=1}^{K} \exp(v_i^\top u_k^s) / u_{ij}^t}
$$

(16)
Further, we simplify by replacing $\exp(v_i^T u_k^j)$ with $\exp(i, jk)$. Equation 16 can now be represented as,

$$p_{ij} = \frac{\sum_{k=1}^{K} \exp(i, jk)}{\sum_{i=1}^{C} \sum_{k'=1}^{K} \exp(i, lk')}$$

(17)

We drop the outer summations (along with the -ve sign) and will reintroduce it at a later time. The entropy loss can be re-phrased using $\log(\frac{a}{b}) = \log(a) - \log(b)$ as,

$$\mathcal{H}_{ij} = \frac{\sum_{k=1}^{K} \exp(i, jk)}{\sum_{i=1}^{C} \sum_{k'=1}^{K} \exp(i, lk')} \log\left( \sum_{k=1}^{K} \exp(i, jk) \right)$$

(18)

$$- \frac{\sum_{k=1}^{K} \exp(i, jk)}{\sum_{i=1}^{C} \sum_{k'=1}^{K} \exp(i, lk')} \log\left( \sum_{i=1}^{C} \sum_{k'=1}^{K} \exp(i, lk') \right)$$

(19)

We need to estimate both, $\frac{\partial \mathcal{H}_{ij}}{\partial v_i}$ for the target and $\frac{\partial \mathcal{H}_{ij}}{\partial u_q^i}$ for the source. We refer to $\partial u_q^i$ for a consistent reference to source data. The derivative $\frac{\partial \mathcal{H}_{ij}}{\partial u_q^i}$ for 18 is,

$$\left[ \frac{\partial \mathcal{H}_{ij}}{\partial u_q^i} \right]_{18} = \frac{v_i}{\sum_{l,k'} \exp(i, lk')} \left[ \sum_k I_{k=q}^{j=p} \exp(i, jk) \log\left( \sum_k \exp(i, jk) \right) + \sum_k I_{k=q}^{j=p} \exp(i, jk) - p_{ij} \exp(i, pq) \log\left( \sum_k \exp(i, jk) \right) \right]$$

(20)

where, $I_{\{\cdot\}}$ is an indicator function which is 1 only when both the conditions within are true, else it is 0. The derivative $\frac{\partial \mathcal{H}_{ij}}{\partial u_q^i}$ for 19 is,

$$\left[ \frac{\partial \mathcal{H}_{ij}}{\partial u_q^i} \right]_{19} = -\frac{v_i}{\sum_{l,k'} \exp(i, lk')} \left[ \sum_k I_{k=q}^{j=p} \exp(i, jk) \log\left( \sum_l \exp(i, lk') \right) + p_{ij} \exp(i, pq) \log\left( \sum_l \exp(i, lk') \right) - p_{ij} \exp(i, pq) \log\left( \sum_k \exp(i, jk) \right) \right]$$

(21)

Expressing $\frac{\partial \mathcal{H}_{ij}}{\partial u_q^i} = \left[ \frac{\partial \mathcal{H}_{ij}}{\partial u_q^i} \right]_{18} + \left[ \frac{\partial \mathcal{H}_{ij}}{\partial u_q^i} \right]_{19}$, and defining $\bar{p}_{ijk} = \exp(i, jk)$ the derivative w.r.t. the source is,

$$\frac{\partial \mathcal{H}_{ij}}{\partial u_q^i} = v_i \left[ \sum_k I_{k=q}^{j=p} \bar{p}_{ijk} \log\left( \sum_k \exp(i, jk) \right) - \sum_k I_{k=q}^{j=p} \bar{p}_{ijk} \log\left( \sum_l \exp(i, lk') \right) - p_{ij} \exp(p, jq) \log\left( \sum_k \exp(i, jk) \right) + p_{ij} \exp(p, jq) \log\left( \sum_l \exp(i, lk') \right) + \sum_k I_{k=q}^{j=p} \bar{p}_{ijk} - p_{ij} \bar{p}_{ipq} \right]$$

(22)

$$= v_i \left[ \sum_k I_{k=q}^{j=p} \bar{p}_{ijk} \log(p_{ij}) - p_{ij} \bar{p}_{ipq} \log(p_{ij}) + \sum_k I_{k=q}^{j=p} \bar{p}_{ijk} - p_{ij} \bar{p}_{ipq} \right]$$

(23)

$$= v_i \left( \log(p_{ij}) + 1 \right) \left[ \sum_k I_{k=q}^{j=p} \bar{p}_{ijk} - p_{ij} \bar{p}_{ipq} \right]$$

(24)

The derivative of $\mathcal{H}$ w.r.t the source output $u_q^i$ is given by,

$$\frac{\partial \mathcal{H}}{\partial u_q^i} = -\frac{1}{n_t} \sum_{t=1}^{n_t} \sum_{j=1}^{C} v_i \left( \log(p_{ij}) + 1 \right) \left[ \sum_k I_{k=q}^{j=p} \bar{p}_{ijk} - p_{ij} \bar{p}_{ipq} \right]$$

(25)

We now outline the derivative $\frac{\partial \mathcal{H}}{\partial v_i}$ for 18 as,

$$\left[ \frac{\partial \mathcal{H}_{ij}}{\partial v_i} \right]_{18} = \frac{1}{\sum_{l,k'} \exp(i, lk')} \left[ \log\left( \sum_k \exp(i, jk) \right) \sum_k \exp(i, jk) u_k^j + \sum_k \exp(i, jk) u_k^j - \sum_{l,k'} \exp(i, lk') \sum_k \exp(i, jk) \log\left( \sum_k \exp(i, jk) \right) \sum_{l,k'} \exp(i, lk') u_k^j \right]$$

(26)
and the derivative $\frac{\partial H_{ij}}{\partial v_i}$ for $19$ as,

$$
\frac{\partial H_{ij}}{\partial v_i} = 19 - \sum_{l,k'} \exp(i, l') \left[ \log\left( \sum_{l,k'} \exp(i, l') \right) \sum_k \exp(i, jk) u_k^l + \frac{\sum_k \exp(i, jk) \sum_{l',k'} \exp(i, l') \sum_{l,k'} \exp(i, l') u_k^{l'} \sum_{l',k'} \exp(i, l')}{\sum_{l,k'} \exp(i, l') \sum_k \exp(i, jk) \log\left( \sum_{l,k'} \exp(i, l') \right) \sum_{l,k'} \exp(i, l') \sum_{l',k'} \exp(i, l') u_k^{l'}}, \right. 

$$

Expressing $\frac{\partial H_{ij}}{\partial v_i} = 18 + \left[ \frac{\partial H_{ij}}{\partial v_i} \right] 19$, we get,

$$
\frac{\partial H_{ij}}{\partial v_i} = \frac{1}{\sum_{l,k'} \exp(i, l')} \left[ \log\left( \sum_{l,k'} \exp(i, jk) \right) \sum_k \exp(i, jk) u_k^l - \log\left( \sum_{l,k'} \exp(i, l') \right) \sum_k \exp(i, jk) u_k^{l'} 

+ \sum_k \exp(i, jk) u_k^l - p_{ij} \sum_{l,k'} \exp(i, l') u_k^{l'} 

- p_{ij} \log\left( \sum_{l,k'} \exp(i, jk) \right) \sum_{l,k'} \exp(i, l') u_k^{l'} + p_{ij} \log\left( \sum_{l,k'} \exp(i, l') \right) \sum_{l,k'} \exp(i, l') u_k^{l'} 

\sum_{l,k'} \exp(i, jk) u_k^l - p_{ij} \sum_{l,k'} \exp(i, l') u_k^{l'} \right]. 

$$

The derivative of $H$ w.r.t. target output $v_q$ is given by,

$$
\frac{\partial H}{\partial v_q} = -\frac{1}{n_t} \sum_{j=1}^C \left( \log(p_{qj}) + 1 \right) \left( \sum_k p_{qjk} u_k^l - p_{qj} \sum_{l,k'} \exp(i, l') u_k^{l'} \right). 

$$

2. Unsupervised Domain Adaptation: Additional Results

In the main paper we had presented results for unsupervised domain adaptation based object recognition with $d = 64$ bits. Here, we outline the classification results with $d = 16$ (DAH-16) and $d = 128$ (DAH-128) bits for the Office-Home dataset in Table 1. We also present the (DAH-64), DAN and DANN results for comparison. There is an increase in the average recognition accuracy for $d = 128$ bits compared to $d = 64$ bits because of the increased capacity in representation. As expected, $d = 16$ has a lower recognition accuracy.

Table 1: Recognition accuracies (%) for domain adaptation experiments on the Office-Home dataset. {Art (Ar), Clipart (Cl), Product (Pr), Real-World (Rw)}. Ar→Cl implies Ar is source and Cl is target.

<table>
<thead>
<tr>
<th>Expt.</th>
<th>Ar→Cl</th>
<th>Ar→Pr</th>
<th>Ar→Rw</th>
<th>Cl→Ar</th>
<th>Cl→Pr</th>
<th>Cl→Rw</th>
<th>Pr→Ar</th>
<th>Pr→Cl</th>
<th>Pr→Rw</th>
<th>Rw→Ar</th>
<th>Rw→Cl</th>
<th>Rw→Pr</th>
<th>Avg.</th>
</tr>
</thead>
<tbody>
<tr>
<td>DAN</td>
<td>30.66</td>
<td>42.17</td>
<td>54.13</td>
<td>32.83</td>
<td>47.59</td>
<td>49.78</td>
<td>29.07</td>
<td>34.05</td>
<td>56.70</td>
<td>43.58</td>
<td>38.25</td>
<td>62.73</td>
<td>43.46</td>
</tr>
<tr>
<td>DANN</td>
<td>33.33</td>
<td>42.96</td>
<td>54.42</td>
<td>32.26</td>
<td>49.13</td>
<td>49.76</td>
<td>30.49</td>
<td>38.14</td>
<td>56.76</td>
<td>44.71</td>
<td>42.66</td>
<td>64.65</td>
<td>44.94</td>
</tr>
<tr>
<td>DAH-16</td>
<td>23.83</td>
<td>30.32</td>
<td>40.14</td>
<td>25.67</td>
<td>38.79</td>
<td>33.26</td>
<td>20.11</td>
<td>27.72</td>
<td>40.90</td>
<td>32.63</td>
<td>25.54</td>
<td>37.46</td>
<td>31.36</td>
</tr>
<tr>
<td>DAH-64</td>
<td>31.64</td>
<td>40.75</td>
<td>51.73</td>
<td>34.69</td>
<td>51.93</td>
<td>52.79</td>
<td>29.91</td>
<td>39.63</td>
<td>60.71</td>
<td>44.99</td>
<td>45.15</td>
<td>62.54</td>
<td>45.54</td>
</tr>
<tr>
<td>DAH-128</td>
<td>32.58</td>
<td>40.64</td>
<td>52.40</td>
<td>35.72</td>
<td>52.80</td>
<td>52.12</td>
<td>30.94</td>
<td>41.31</td>
<td>59.31</td>
<td>45.65</td>
<td>46.67</td>
<td>64.97</td>
<td>46.26</td>
</tr>
</tbody>
</table>

3. Unsupervised Domain Adaptive Hashing: Additional Results

We provide the unsupervised domain adaptive hashing results for $d = 16$ and $d = 128$ bits in Figures 1 and 2 respectively. In Tables 2 and 3, we outline the corresponding mAP values. The notations are along the lines outlined in the main paper. We observe similar trends for both $d = 16$ and $d = 128$ bits compared to $d = 64$ bits. It is interesting to note that with increase in bit size $d$, the mAP does not necessarily increase. Table 3 ($d = 64$) has its mAP values lower than those for $d = 64$ (see main paper) for all the hashing methods. This indicates that merely increasing the hash code length does not always improve mAP scores. Also, the mAP values for Real-World for $d = 128$ bits has DAH performing better than SuH. This indicates that in some cases domain adaptation helps in learning a better generalized model.
Table 2: Mean average precision @16 bits. For the NoDA and DAH results, Art is the source domain for Clipart, Product and Real-World and Clipart is the source domain for Art.

<table>
<thead>
<tr>
<th>Expt.</th>
<th>NoDA</th>
<th>ITQ</th>
<th>KMeans</th>
<th>BA</th>
<th>BDNN</th>
<th>DAH</th>
<th>SuH</th>
</tr>
</thead>
<tbody>
<tr>
<td>Art</td>
<td>0.102</td>
<td>0.147</td>
<td>0.133</td>
<td>0.131</td>
<td>0.151</td>
<td>0.207</td>
<td>0.381</td>
</tr>
<tr>
<td>Clipart</td>
<td>0.110</td>
<td>0.120</td>
<td>0.116</td>
<td>0.123</td>
<td>0.138</td>
<td>0.211</td>
<td>0.412</td>
</tr>
<tr>
<td>Product</td>
<td>0.134</td>
<td>0.253</td>
<td>0.241</td>
<td>0.253</td>
<td>0.313</td>
<td>0.257</td>
<td>0.459</td>
</tr>
<tr>
<td>Real-World</td>
<td>0.193</td>
<td>0.225</td>
<td>0.195</td>
<td>0.216</td>
<td>0.248</td>
<td>0.371</td>
<td>0.400</td>
</tr>
<tr>
<td>Avg.</td>
<td>0.135</td>
<td>0.186</td>
<td>0.171</td>
<td>0.181</td>
<td>0.212</td>
<td>0.262</td>
<td>0.413</td>
</tr>
</tbody>
</table>

Table 3: Mean average precision @128 bits. For the NoDA and DAH results, Art is the source domain for Clipart, Product and Real-World and Clipart is the source domain for Art.

<table>
<thead>
<tr>
<th>Expt.</th>
<th>NoDA</th>
<th>ITQ</th>
<th>KMeans</th>
<th>BA</th>
<th>BDNN</th>
<th>DAH</th>
<th>SuH</th>
</tr>
</thead>
<tbody>
<tr>
<td>Art</td>
<td>0.154</td>
<td>0.202</td>
<td>0.175</td>
<td>0.148</td>
<td>0.207</td>
<td>0.314</td>
<td>0.444</td>
</tr>
<tr>
<td>Clipart</td>
<td>0.186</td>
<td>0.210</td>
<td>0.196</td>
<td>0.187</td>
<td>0.213</td>
<td>0.350</td>
<td>0.346</td>
</tr>
<tr>
<td>Product</td>
<td>0.279</td>
<td>0.416</td>
<td>0.356</td>
<td>0.336</td>
<td>0.432</td>
<td>0.424</td>
<td>0.792</td>
</tr>
<tr>
<td>Real-World</td>
<td>0.308</td>
<td>0.343</td>
<td>0.289</td>
<td>0.258</td>
<td>0.348</td>
<td>0.544</td>
<td>0.458</td>
</tr>
<tr>
<td>Avg.</td>
<td>0.232</td>
<td>0.293</td>
<td>0.254</td>
<td>0.232</td>
<td>0.300</td>
<td>0.408</td>
<td>0.510</td>
</tr>
</tbody>
</table>

Figure 1: Precision-Recall curves @16 bits for the Office-Home dataset. Comparison of hashing without domain adaptation (NoDA), shallow unsupervised hashing (ITQ, KMeans), state-of-the-art deep unsupervised hashing (BA, BDNN), unsupervised domain adaptive hashing (DAH) and supervised hashing (SuH). Best viewed in color.

Figure 2: Precision-Recall curves @128 bits for the Office-Home dataset. Comparison of hashing without domain adaptation (NoDA), shallow unsupervised hashing (ITQ, KMeans), state-of-the-art deep unsupervised hashing (BA, BDNN), unsupervised domain adaptive hashing (DAH) and supervised hashing (SuH). Best viewed in color.

References