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;

$$\min_{\mathcal{U}} \mathcal{J} = \mathcal{L}(\mathcal{U}_s) + \gamma\mathcal{M}(\mathcal{U}_s, \mathcal{U}_t) + \eta\mathcal{H}(\mathcal{U}_s, \mathcal{U}_t), \tag{8}$$

where, $\mathcal{U} := \{\mathcal{U}_s \cup \mathcal{U}_t\}$ and (γ, η) 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 $\mathcal{U}$.

1.1. Derivative for MK-MMD

$$\mathcal{M}(\mathcal{U}_s, \mathcal{U}_t) = \sum_{l\in\mathcal{F}} d_k^2(\mathcal{U}_s^l, \mathcal{U}_t^l), \tag{1}$$

$$d_k^2(\mathcal{U}_s^l, \mathcal{U}_t^l) = \Big|\Big|\mathbb{E}[\phi(\boldsymbol{u}^{s,l})] - \mathbb{E}[\phi(\boldsymbol{u}^{t,l})]\Big|\Big|^2_{\mathscr{H}_k}. \tag{2}$$

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 $\boldsymbol{u}_i$ and the output of the i^{th} target data point is represented as $\boldsymbol{v}_i$. 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 $\mathcal{O}(n^2)$, the MK-MMD loss outlined in [1] is $\mathcal{O}(n)$ 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 $\mathcal{U} := \{\boldsymbol{u}_i\}_{i=1}^n$ and the number of target data points $\mathcal{V} := \{\boldsymbol{v}_i\}_{i=1}^n$ 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 $\boldsymbol{w}_i = [\boldsymbol{u}_{2i-1}, \boldsymbol{u}_{2i}, \boldsymbol{v}_{2i-1}, \boldsymbol{v}_{2i}], \forall i \in \{1, 2, \ldots, n/2\}$. The MK-MMD is given by,

$$\mathcal{M}(\mathcal{U}, \mathcal{V}) = \sum_{m=1}^{\kappa} \beta_m \frac{1}{n/2} \sum_{i=1}^{n/2} h_m(\boldsymbol{w}_i), \tag{9}$$

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

$$h_m(\boldsymbol{w}_i) = k_m(\boldsymbol{u}_{2i-1}, \boldsymbol{u}_{2i}) + k_m(\boldsymbol{v}_{2i-1}, \boldsymbol{v}_{2i}) - k_m(\boldsymbol{u}_{2i-1}, \boldsymbol{v}_{2i}) - k_m(\boldsymbol{u}_{2i}, \boldsymbol{v}_{2i-1}), \tag{10}$$

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

$$\mathcal{M}(\mathcal{U}, \mathcal{V}) = \frac{2}{n\kappa} \sum_{m=1}^{\kappa} \sum_{i=1}^{n/2} \big[k_m(\boldsymbol{u}_{2i-1}, \boldsymbol{u}_{2i}) + k_m(\boldsymbol{v}_{2i-1}, \boldsymbol{v}_{2i}) - k_m(\boldsymbol{u}_{2i-1}, \boldsymbol{v}_{2i}) - k_m(\boldsymbol{u}_{2i}, \boldsymbol{v}_{2i-1})\big], \tag{11}$$

We now outline the derivative of 11 w.r.t. source output $\boldsymbol{u}_q$ and target output $\boldsymbol{v}_q$. The derivative is,

$$\frac{\partial \mathcal{M}}{\partial \boldsymbol{u}_q} = \frac{2}{n\kappa}\sum_{m=1}^{\kappa}\sum_{i=1}^{n/2}\Big[\frac{2}{\sigma_m}k_m(\boldsymbol{u}_{2i-1},\boldsymbol{u}_{2i}).(\boldsymbol{u}_{2i-1}-\boldsymbol{u}_{2i}).(\mathcal{I}\{q=2i\}-\mathcal{I}\{q=2i-1\})$$
$$+\frac{2}{\sigma_m}k_m(\boldsymbol{u}_{2i-1},\boldsymbol{v}_{2i}).(\boldsymbol{u}_{2i-1}-\boldsymbol{v}_{2i}).\mathcal{I}\{q=2i-1\}+\frac{2}{\sigma_m}k_m(\boldsymbol{u}_{2i},\boldsymbol{v}_{2i-1}).(\boldsymbol{u}_{2i}-\boldsymbol{v}_{2i-1}).\mathcal{I}\{q=2i\}\Big], \tag{12}$$

where, $\mathcal{I}\{.\}$ 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 $\boldsymbol{v}_q$ is,

$$\frac{\partial \mathcal{M}}{\partial \boldsymbol{v}_q} = \frac{2}{n\kappa}\sum_{m=1}^{\kappa}\sum_{i=1}^{n/2}\Big[\frac{2}{\sigma_m}k_m(\boldsymbol{v}_{2i-1},\boldsymbol{v}_{2i}).(\boldsymbol{v}_{2i-1}-\boldsymbol{v}_{2i}).(\mathcal{I}\{q=2i\}-\mathcal{I}\{q=2i-1\})$$
$$-\frac{2}{\sigma_m}k_m(\boldsymbol{u}_{2i-1},\boldsymbol{v}_{2i}).(\boldsymbol{u}_{2i-1}-\boldsymbol{v}_{2i}).\mathcal{I}\{q=2i\}-\frac{2}{\sigma_m}k_m(\boldsymbol{u}_{2i},\boldsymbol{v}_{2i-1}).(\boldsymbol{u}_{2i}-\boldsymbol{v}_{2i-1}).\mathcal{I}\{q=2i-1\}\Big], \tag{13}$$

1.2. Derivative for Supervised Hash Loss

The supervised hash loss is given by,

$$\min_{\mathcal{U}_s}\mathcal{L}(\mathcal{U}_s) = -\sum_{s_{ij}\in\mathcal{S}}\Big(s_{ij}\boldsymbol{u}_i^\top\boldsymbol{u}_j - \log\big(1+\exp(\boldsymbol{u}_i^\top\boldsymbol{u}_j)\big)\Big)$$
$$+\sum_{i=1}^{n_s}\big|\big|\boldsymbol{u}_i - \text{sgn}(\boldsymbol{u}_i)\big|\big|_2^2. \tag{5}$$

The partial derivative of 5 w.r.t. source data output $\boldsymbol{u}_p$ is given by,

$$\frac{\partial \mathcal{L}}{\partial \boldsymbol{u}_q} = \sum_{s_{ij}\in\mathcal{S}}\Big[I\{i=q\}\big(\sigma(\boldsymbol{u}_i^\top\boldsymbol{u}_j)-s_{ij}\big)\boldsymbol{u}_j + I\{j=q\}\big(\sigma(\boldsymbol{u}_i^\top\boldsymbol{u}_j)-s_{ij}\big)\boldsymbol{u}_i\Big] + 2(\boldsymbol{u}_q - \text{sgn}(\boldsymbol{u}_q)) \tag{14}$$

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

$$\frac{\partial \mathcal{L}}{\partial \boldsymbol{u}_q} = \sum_{j=1}^{n_s}\Big[2\big(\sigma(\boldsymbol{u}_q^\top\boldsymbol{u}_j)-s_{qj}\big)\boldsymbol{u}_j\Big] + 2\big(\boldsymbol{u}_q - \text{sgn}(\boldsymbol{u}_q)\big). \tag{15}$$

1.3. Derivative for Unsupervised Entropy Loss

We outline the derivative of $\frac{d\mathcal{H}}{d\mathcal{U}}$ in the following section, where $\mathcal{H}$ is defined as,

$$\mathcal{H}(\mathcal{U}_s,\mathcal{U}_t) = -\frac{1}{n_t}\sum_{i=1}^{n_t}\sum_{j=1}^{C}p_{ij}\log(p_{ij}) \tag{7}$$

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

$$p_{ij} = \frac{\sum_{k=1}^{K}\exp(\boldsymbol{u}_i^{t\top}\boldsymbol{u}_k^{s_j})}{\sum_{l=1}^{C}\sum_{k'=1}^{K}\exp(\boldsymbol{u}_i^{t\top}\boldsymbol{u}_{k'}^{s_l})} \tag{6}$$

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

$$p_{ij} = \frac{\sum_{k=1}^{K}\exp(\boldsymbol{v}_i{}^\top\boldsymbol{u}_k^j)}{\sum_{l=1}^{C}\sum_{k'=1}^{K}\exp(\boldsymbol{v}_i{}^\top\boldsymbol{u}_{k'}^l)} \tag{16}$$

Further, we simplify by replacing exp($\boldsymbol{v}_i^\top \boldsymbol{u}_k^j$) with exp$(i, jk)$. Equation 16 can now be represented as,

$$p_{ij} = \frac{\sum_{k=1}^{K} \text{exp}(i, jk)}{\sum_{l=1}^{C} \sum_{k'=1}^{K} \text{exp}(i, lk')} \tag{17}$$

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,

$$\mathcal{H}_{ij} = \frac{\sum_{k=1}^{K} \text{exp}(i, jk)}{\sum_{l=1}^{C} \sum_{k'=1}^{K} \text{exp}(i, lk')} \text{log}\big(\textstyle\sum_{k=1}^{K} \text{exp}(i, jk)\big) \tag{18}$$

$$- \frac{\sum_{k=1}^{K} \text{exp}(i, jk)}{\sum_{l=1}^{C} \sum_{k'=1}^{K} \text{exp}(i, lk')} \text{log}\big(\textstyle\sum_{l=1}^{C} \sum_{k'=1}^{K} \text{exp}(i, lk')\big) \tag{19}$$

We need to estimate both, $\frac{\partial \mathcal{H}_{ij}}{\partial \boldsymbol{v}_i}$ for the target and $\frac{\partial \mathcal{H}_{ij}}{\partial \boldsymbol{u}_q^p}$ for the source. We refer to $\partial \boldsymbol{u}_q^p$ for a consistent reference to source data. The derivative $\frac{\partial \mathcal{H}_{ij}}{\partial \boldsymbol{u}_q^p}$ for 18 is,

$$\left[\frac{\partial \mathcal{H}_{ij}}{\partial \boldsymbol{u}_q^p}\right]_{18} = \frac{\boldsymbol{v}_i}{\sum_{l,k'} \text{exp}(i, lk')} \Big[\textstyle\sum_k I\{^{j=p,}_{k=q}\} \text{exp}(i, jk).\text{log}\big(\sum_k \text{exp}(i, jk)\big) + \sum_k I\{^{j=p,}_{k=q}\} \text{exp}(i, jk)$$

$$- p_{ij} \text{exp}(i, pq) \text{log}\big(\textstyle\sum_k \text{exp}(i, jk)\big)\Big], \tag{20}$$

where, $I\{.\}$ is an indicator function which is 1 only when both the conditions within are true, else it is 0. The derivative $\frac{\partial \mathcal{H}_{ij}}{\partial \boldsymbol{u}_q^p}$ for 19 is,

$$\left[\frac{\partial \mathcal{H}_{ij}}{\partial \boldsymbol{u}_q^p}\right]_{19} = -\frac{\boldsymbol{v}_i}{\sum_{l,k'} \text{exp}(i, lk')} \Big[\textstyle\sum_k I\{^{j=p,}_{k=q}\} \text{exp}(i, jk).\text{log}\big(\sum_{l,k'} \text{exp}(i, lk')\big) + p_{ij} \text{exp}(i, pq)$$

$$- p_{ij} \text{exp}(i, pq) \text{log}\big(\textstyle\sum_{l,k'} \text{exp}(i, lk')\big)\Big] \tag{21}$$

Expressing $\frac{\partial \mathcal{H}_{ij}}{\partial \boldsymbol{u}_q^p} = \left[\frac{\partial \mathcal{H}_{ij}}{\partial \boldsymbol{u}_q^p}\right]_{18} + \left[\frac{\partial \mathcal{H}_{ij}}{\partial \boldsymbol{u}_q^p}\right]_{19}$, and defining $\bar{p}_{ijk} = \frac{\text{exp}(i,jk)}{\sum_{l,k'} \text{exp}(i,lk')}$ the derivative w.r.t. the source is,

$$\frac{\partial \mathcal{H}_{ij}}{\partial \boldsymbol{u}_q^p} = \boldsymbol{v}_i \Big[\textstyle\sum_k I\{^{j=p,}_{k=q}\} \bar{p}_{ijk}.\text{log}\big(\sum_k \text{exp}(i, jk)\big) + \sum_k I\{^{j=p,}_{k=q}\} \bar{p}_{ijk}$$

$$- p_{ij} \bar{p}_{ipq} \text{log}\big(\textstyle\sum_k \text{exp}(i, jk)\big) - \sum_k I\{^{j=p,}_{k=q}\} \bar{p}_{ijk}.\text{log}\big(\sum_{l,k'} \text{exp}(i, lk')\big)$$

$$- p_{ij} \bar{p}_{ipq} + p_{ij} \bar{p}_{ipq} \text{log}\big(\textstyle\sum_{l,k'} \text{exp}(i, lk')\big)\Big] \tag{22}$$

$$= \boldsymbol{v}_i \Big[\textstyle\sum_k I\{^{j=p,}_{k=q}\} \bar{p}_{ijk} \text{log}(p_{ij}) - p_{ij} \bar{p}_{ipq} \text{log}(p_{ij}) + \sum_k I\{^{j=p,}_{k=q}\} \bar{p}_{ijk} - p_{ij} \bar{p}_{ipq}\Big] \tag{23}$$

$$= \boldsymbol{v}_i \big(\text{log}(p_{ij}) + 1\big) \Big[\textstyle\sum_k I\{^{j=p,}_{k=q}\} \bar{p}_{ijk} - p_{ij} \bar{p}_{ipq}\Big] \tag{24}$$

The derivative of $\mathcal{H}$ w.r.t the **source** output $\boldsymbol{u}_q^p$ is given by,

$$\frac{\partial \mathcal{H}}{\partial \boldsymbol{u}_q^p} = -\frac{1}{n_t} \sum_{i=1}^{n_t} \sum_{j=1}^{C} \boldsymbol{v}_i \big(\text{log}(p_{ij}) + 1\big) \Big[\textstyle\sum_k I\{^{j=p,}_{k=q}\} \bar{p}_{ijk} - p_{ij} \bar{p}_{ipq}\Big] \tag{25}$$

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

$$\left[\frac{\partial \mathcal{H}_{ij}}{\partial \boldsymbol{v}_i}\right]_{18} = \frac{1}{\sum_{l,k'} \text{exp}(i, lk')} \Big[\text{log}\big(\textstyle\sum_k \text{exp}(i, jk)\big) \sum_k \text{exp}(i, jk) \boldsymbol{u}_k^j + \sum_k \text{exp}(i, jk) \boldsymbol{u}_k^j$$

$$- \frac{1}{\sum_{l,k'} \text{exp}(i, lk')} \textstyle\sum_k \text{exp}(i, jk) \text{log}\big(\sum_k \text{exp}(i, jk)\big) \sum_{l,k'} \text{exp}(i, lk') \boldsymbol{u}_{k'}^l\Big], \tag{26}$$

and the derivative $\frac{\partial \mathcal{H}}{\partial \boldsymbol{v}_i}$ for 19 as,

$$\Big[\frac{\partial \mathcal{H}_{ij}}{\partial \boldsymbol{v}_i}\Big]_{19} = -\frac{1}{\sum_{l,k'} \exp(i,lk')}\Big[\log\big(\sum_{l,k'} \exp(i,lk')\big)\sum_k \exp(i,jk)\boldsymbol{u}_k^j + \frac{\sum_k \exp(i,jk)}{\sum_{l,k'}\exp(i,lk')}\sum_{l,k'} \exp(i,lk')\boldsymbol{u}_{k'}^l \\ - \frac{1}{\sum_{l,k'} \exp(i,lk')}\sum_k \exp(i,jk)\log\big(\sum_{l,k'} \exp(i,lk')\big)\sum_{l,k'} \exp(i,lk')\boldsymbol{u}_{k'}^l\Big], \quad (27)$$

Expressing $\frac{\partial \mathcal{H}_{ij}}{\partial \boldsymbol{v}_i} = \Big[\frac{\partial \mathcal{H}_{ij}}{\partial \boldsymbol{v}_i}\Big]_{18} + \Big[\frac{\partial \mathcal{H}_{ij}}{\partial \boldsymbol{v}_i}\Big]_{19}$, we get,

$$\frac{\partial \mathcal{H}_{ij}}{\partial \boldsymbol{v}_i} = \frac{1}{\sum_{l,k'} \exp(i,lk')}\Big[\log\big(\sum_k \exp(i,jk)\big)\sum_k \exp(i,jk)\boldsymbol{u}_k^j - \log\big(\sum_{l,k'} \exp(i,lk')\big)\sum_k \exp(i,jk)\boldsymbol{u}_k^j \\ + \sum_k \exp(i,jk)\boldsymbol{u}_k^j - p_{ij}\sum_{l,k'} \exp(i,lk')\boldsymbol{u}_{k'}^l \\ - p_{ij}\log\big(\sum_k \exp(i,jk)\big)\sum_{l,k'} \exp(i,lk')\boldsymbol{u}_{k'}^l + p_{ij}\log\big(\sum_{l,k'} \exp(i,lk')\big)\sum_{l,k'} \exp(i,lk')\boldsymbol{u}_{k'}^l\Big] \quad (28)$$

$$= \Big[\log\big(\sum_k \exp(i,jk)\big)\sum_k \bar{p}_{ijk}\boldsymbol{u}_k^j - \log\big(\sum_{l,k'} \exp(i,lk')\big)\sum_k \bar{p}_{ijk}\boldsymbol{u}_k^j \\ + \sum_k \bar{p}_{ijk}\boldsymbol{u}_k^j - p_{ij}\sum_{l,k'} \bar{p}_{ijk'}\boldsymbol{u}_{k'}^l \\ - p_{ij}\log\big(\sum_k \exp(i,jk)\big)\sum_{l,k'} \bar{p}_{ijk'}\boldsymbol{u}_{k'}^l + p_{ij}\log\big(\sum_{l,k'} \exp(i,lk')\big)\sum_{l,k'} \bar{p}_{ijk'}\boldsymbol{u}_{k'}^l\Big] \quad (29)$$

$$= \big(\log(p_{ij}) + 1\big)\sum_k \bar{p}_{ijk}\boldsymbol{u}_k^j - \big(\log(p_{ij}) + 1\big)p_{ij}\sum_{l,k'} \bar{p}_{ijk'}\boldsymbol{u}_{k'}^l \quad (30)$$

$$= \big(\log(p_{ij}) + 1\big)\big(\sum_k \bar{p}_{ijk}\boldsymbol{u}_k^j - p_{ij}\sum_{l,k'} \bar{p}_{ijk'}\boldsymbol{u}_{k'}^l\big) \quad (31)$$

The derivative of $\mathcal{H}$ w.r.t. **target** output $\boldsymbol{v}_q$ is given by,

$$\frac{\partial \mathcal{H}}{\partial \boldsymbol{v}_q} = -\frac{1}{n_t}\sum_{j=1}^{C}\big(\log(p_{qj}) + 1\big)\big(\sum_k \bar{p}_{qjk}\boldsymbol{u}_k^j - p_{qj}\sum_{l,k'} \bar{p}_{qjk'}\boldsymbol{u}_{k'}^l\big) \quad (32)$$

The derivative of $\mathcal{H}$ w.r.t. the source outputs is given by 25 and w.r.t. the target outputs is given by 32.

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.

Expt.	Ar→Cl	Ar→Pr	Ar→Rw	Cl→Ar	Cl→Pr	Cl→Rw	Pr→Ar	Pr→Cl	Pr→Rw	Rw→Ar	Rw→Cl	Rw→Pr	Avg.
DAN	30.66	42.17	54.13	32.83	47.59	49.78	29.07	34.05	56.70	43.58	38.25	62.73	43.46
DANN	33.33	42.96	54.42	32.26	49.13	49.76	30.49	38.14	56.76	44.71	42.66	64.65	44.94
DAH-16	23.83	30.32	40.14	25.67	38.79	33.26	20.11	27.72	40.90	32.63	25.54	37.46	31.36
DAH-64	31.64	40.75	51.73	34.69	51.93	52.79	29.91	39.63	60.71	44.99	45.13	62.54	45.54
DAH-128	32.58	40.64	52.40	35.72	52.80	52.12	30.94	41.31	59.31	45.65	46.67	64.97	46.26

3. Unsupervised Domain Adaptive Hashing: Additional Results

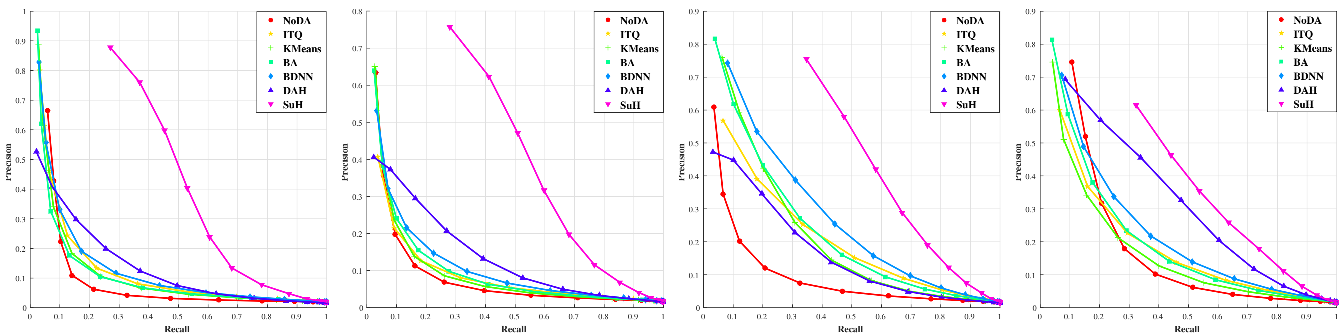
We provide the unsupervised domain adaptive hashing results for $d = 16$ and $d = 128$ bits in Figures 1 and 2 respectively. In Tables 2 and 3, we outline the corresponding mAP values. The notations are along the lines outlined in the main paper. We observe similar trends for both $d = 16$ and $d = 128$ 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`.

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

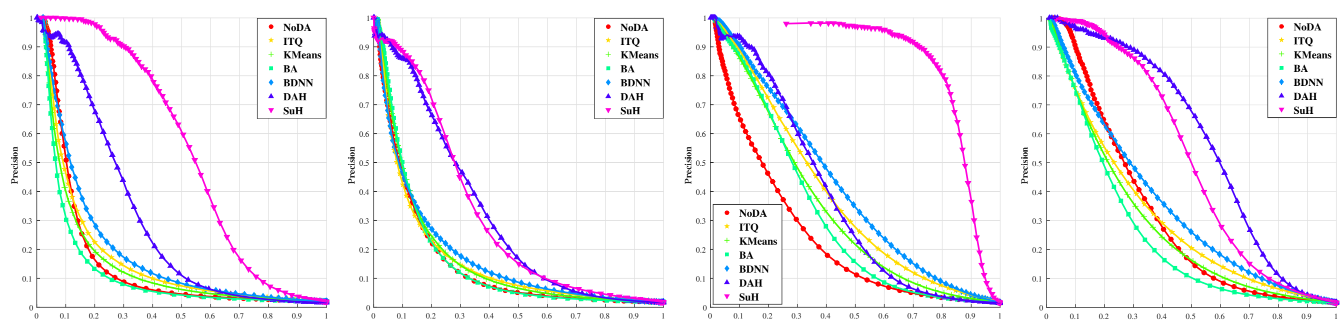
Table 3: Mean average precision @128 bits. For the NoDA and DAH results, `Art` is the source domain for `Clipart`, `Product` and `Real-World` and `Clipart` is the source domain for `Art`.

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



(a) Art **(b)** Clipart **(c)** Product **(d)** Real-World

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.



(a) Art **(b)** Clipart **(c)** Product **(d)** Real-World

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

[1] A. Gretton, D. Sejdinovic, H. Strathmann, S. Balakrishnan, M. Pontil, K. Fukumizu, and B. K. Sriperumbudur. Optimal kernel choice for large-scale two-sample tests. In *Advances in neural information processing systems*, pages 1205–1213, 2012. 1