Supplementary Material for
Drop to Adapt:
Learning Discriminative Features for Unsupervised Domain Adaptation

Seungmin Lee*
Seoul National Univ.

Dongwan Kim*
Seoul National Univ.

Namil Kim
NA VER LABS

Seong-Gyun Jeong
CODE42.ai

In this supplementary material, we derive an approximation of the channel-wise adversarial dropout ($\S$ Appendix A) and provide implementation details of the experiments ($\S$ Appendix B). Lastly, we provide additional GradCAM visualizations ($\S$ Appendix C).

Appendix A. Approximation of Channel-wise Adversarial Dropout

Without loss of generality, the dropout mask $m$ is vectorized to $v = vec(m) \in \mathbb{R}^{CHW}$. Similarly, $v^0$ and $v^s$ represent vectorized forms of $m^0$ and $m^s$, respectively. After vectorization of $m$, we refer to the elements of $m(i)$ with a set of indices $\pi_i$, and impose the channel-wise dropout constraints as follows:

$$v[\pi_i] = vec(m(i)) = 0 \text{ or } 1 \in \mathbb{R}^{HW}.\quad (a-1)$$

Let denote $d(x, v; v^s) = D[h(x; v^s), h(x; v)]$ as the divergence between two outputs using different dropout masks for convenience sake. Assuming $d$ is a differentiable function with respect to $v$, it can be approximated by a first-order Taylor expansion:

$$d(x, v; v^s) \approx d(x, v^0; v^s) + (v - v^0)^T J \mid_{v=v^0}.\quad (a-2)$$

This equation shows that the Jacobian is proportional to the divergence. In other words,

$$d(x, v; v^s) \propto v^T J. \quad (a-2)$$

We now see that the elements of $J$ correspond to the impact values, which indicate the contribution of each activation over the divergence metric. Thus, for the given Jacobian, we can systematically modify the elements of $v$ to maximize the divergence. However, due to the channel-wise dropout constraint from Eq. $\S$, we cannot modify each element individually. Instead, we reformulate the above relationship as:

$$d(x, v; v^s) \propto \sum_i C v[\pi_i]^T J[\pi_i].\quad (a-3)$$

The impact value $s$ of the $i$-th activation map in $h_l(x)$ can be defined as:

$$s_i = 1^T J[\pi_i]. \quad (a-4)$$

Consequently, after computing the impact values $s$, we solve 0/1 Knapsack problem as proposed in [3] while holding the constraints (a-1).

* denotes equal contribution.

This work was done while the authors were at NAVER LABS.
Table A-1. Hyperparameters

<table>
<thead>
<tr>
<th>Experiment</th>
<th>Backbone</th>
<th>(\lambda_1)</th>
<th>(\lambda_2)</th>
<th>(\lambda_3)</th>
<th>(\delta_c)</th>
<th>(\delta_c)</th>
<th>(T_r)</th>
<th>(\epsilon)</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Small dataset</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SVHN (\rightarrow) MNIST</td>
<td>9 Conv+1 FC</td>
<td>2</td>
<td>0.01</td>
<td>0.1</td>
<td>0.05</td>
<td>80</td>
<td>3.5</td>
<td></td>
</tr>
<tr>
<td>MNIST (\rightarrow) USPS</td>
<td>3 Conv+2 FC</td>
<td>2</td>
<td>0.01</td>
<td>0</td>
<td>0.05</td>
<td>80</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>USPS (\rightarrow) MNIST</td>
<td>3 Conv+2 FC</td>
<td>2</td>
<td>0.01</td>
<td>0.1</td>
<td>0.05</td>
<td>80</td>
<td>3.5</td>
<td></td>
</tr>
<tr>
<td>STL (\rightarrow) CIFAR</td>
<td>9 Conv+1 FC</td>
<td>2</td>
<td>0.01</td>
<td>0.1</td>
<td>0</td>
<td>80</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>CIFAR (\rightarrow) STL</td>
<td>9 Conv+1 FC</td>
<td>2</td>
<td>0.01</td>
<td>0</td>
<td>0</td>
<td>80</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td><strong>Large dataset</strong></td>
<td>ResNet-50</td>
<td>2</td>
<td>0.02</td>
<td>0.2</td>
<td>0.1</td>
<td>0.01</td>
<td>20</td>
<td>15</td>
</tr>
<tr>
<td>VisDA-2017 Classification</td>
<td>ResNet-101</td>
<td>2</td>
<td>0.02</td>
<td>0.2</td>
<td>0.1</td>
<td>0.01</td>
<td>30</td>
<td>15</td>
</tr>
<tr>
<td><strong>Semantic segmentation</strong></td>
<td>GTA5 (\rightarrow) Cityscapes</td>
<td>2</td>
<td>0.01</td>
<td>0</td>
<td>0</td>
<td>0.02</td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>

Appendix B. Implementation Details

Training with DTA Loss

We apply a ramp-up factor on DTA loss function \(L_{DTA}\) to stabilize the training process. Instead of directly modulating the weight term \(\lambda_1\), we gradually increase the perturbation magnitudes \(\delta_e\) and \(\delta_c\) which decide the number of hidden units to be eliminated. It allows us to regulate the consistency term, and to train the network being robust to various levels of perturbation generated by the adversarial dropout. We update the ramp-up factors with the following schedule:

\[
\beta^{(t)} = \min(1, \frac{t}{T_r}),
\]

where \(T_r\) represents the ramp-up period, and \(\beta^{(t)}\) denotes the ramp up factor at the current epoch \(t\). Finally, the perturbation magnitude is defined as:

\[
\delta^{(t)} = \beta^{(t)} \delta_e,
\]

where \(\delta\) denotes the maximum level of perturbation. In practice, the same ramp-up period \(T_r\) is applied for both \(\delta_e\) and \(\delta_c\).

Hyperparameters

Table A-1 presents the hyperparameters used in our experiments. We followed a similar hyperparameter search protocol as Shu et al. [5], where we sample a very small subset of labels from the target domain training set. For each objective function, we limit the hyperparameter search to a predefined set of values: \(\lambda_1 = \{2\}, \lambda_2 = \{0, 0.01, 0.02\}, \lambda_3 = \{0, 0.1, 0.2\}, \delta_e = \{0, 0.1\}, \delta_c = \{0, 0.01, 0.02, 0.05\}\), and \(\epsilon = \{0, 3.5, 15\}\). Furthermore, we provide the rest of parameters related to network training for each experimental set up.

Small dataset. All small dataset experiments were trained for 90 epochs, using Adam optimizer [11] with an initial learning rate of 0.001, decaying by a factor of 0.1 every 30 epochs.

Large dataset. We conducted the VisDA-2017 classification experiments on ResNet-50 and ResNet-101. We trained the networks for 20 epochs using Stochastic Gradient Descent (SGD) with a momentum value of 0.9 and an initial learning rate of 0.001, which decays by a factor of 0.1 after 10th epoch.

Semantic segmentation. The semantic segmentation task for domain adaptation from GTA5 to Cityscapes was trained for 5 epochs using SGD with a momentum of 0.9. Since FCN [22] has no fully-connected layers, \(\delta_e\) was automatically set to 0. In addition, we used the maximum \(\delta_e\) value from the beginning because the task-specific objective were dominant in the early stages of training. In this experiment, we turned off VAT objective which hinders from learning the segmentation task.
Appendix C. Additional GradCAM visualizations

In Figure A-1 we provide additional GradCAM visualizations to highlight the effects of adversarial dropout.

![GradCAM visualizations](image)

Figure A-1. Effect of adversarial dropout, visualized by GradCAM.

References