Learning Transferrable Representations for Unsupervised Domain Adaptation
- Supplementary Material

Ozan Sener¹, Hyun Oh Song¹, Ashutosh Saxena², Silvio Savarese¹
Stanford University¹, Brain of Things²
{ozan,hsong,asaxena,ssilvio}@cs.stanford.edu

Abstract

In this supplementary material, we study the sensitivity of the parameter $M$ and give details on the computation of sub-gradient of the loss function with respect to feature weights $\theta_c, \theta_s, \theta_t$. Cost of the reject option $\gamma$ is function of the parameter $M$ and we show that, it is possible to obtain high accuracy over a wide range of $M$ values.

1 Sensitivity Analysis of $M$

As we explained in the main paper, we use the following update function to control the reject cost during training: $\gamma = 0.1 + \frac{\#\text{epoch} - 1}{M}$. In order to analyze the sensitivity of our algorithm on $M$, we swept the scaling factor $(M)$ between 0 and 160 in the SVHN→MNIST experiment. We computed accuracy for each value of $M$ after the adaptation is completed. We display the resulting accuracy vs $M$ plot in Figure 1.

![Accuracy vs M](image)

Figure 1: Effect of reject cost $\gamma$- on the accuracy. Please note that there is large range of $M$ values yielding high accuracy. Hence, our model is not sensitive to the $M$ parameter.

Figure 1 shows that there is a large range of values for $M$ which results in high accuracy transduction. Moreover, for very-large values of $M$, the reject cost $\gamma$ will almost always be low during the training preventing transduction algorithm to make any prediction other than reject. Hence, we expect this
set-up to perform similarly to source only baseline since adaptation is not possible when all labels are reject. Whereas, for very-small values of $M$, the reject cost $\gamma$ will immediately be very-large and our transduction method will not be able to predict reject at all. Hence, we expect this setup to perform similarly to no reject baseline. Indeed, we see this asymptotic behavior and the extreme values of the plot is compatible with numbers in Table 2 of the main paper.

2 Sub-gradients of the loss function for adaptation

We need the explicit form of feature functions to compute the sub-gradients. With a slight abuse of notation, we define the following functions $\Phi_s(\cdot) = \phi_s(\phi_c(\cdot))$ and $\Phi_t(\cdot) = \phi_t(\phi_c(\cdot))$ such that $\phi_s$, $\phi_t$ and $\phi_c$ are functions of $\theta_s$, $\theta_t$ and $\theta_c$ respectively.

Using these terms, we can re-write the loss function from (main paper 3) as;

$$\text{loss} = \sum_{i \in [N_s]} [\phi_s(\phi_c(\hat{x}_i))^T \phi_t(\phi_c(x_{i-})) - \phi_s(\phi_c(\hat{x}_i))^T \phi_t(\phi_c(x_{i+})) + \alpha]_+ \tag{1}$$

The sub-gradients can then further be computed as;

$$\frac{\partial \text{loss}}{\partial \theta_s} = \sum_{i \in [N_s]} 1(\Phi_s(\hat{x}_i)^T \Phi_t(x_{i+}) - \Phi_s(\hat{x}_i)^T \Phi_t(x_{i-}) < \alpha) \left[ \frac{\partial \phi_s}{\partial \phi_c}_{\left| \phi_c(\hat{x}_i) \right.} \left( \Phi_t(x_{i-}) - \Phi_t(x_{i+}) \right) \right]$$

$$\frac{\partial \text{loss}}{\partial \theta_t} = \sum_{i \in [N_s]} 1(\Phi_s(\hat{x}_i)^T \Phi_t(x_{i+}) - \Phi_s(\hat{x}_i)^T \Phi_t(x_{i-}) < \alpha) \left[ \left( \frac{\partial \phi_t}{\partial \phi_c}_{\left| \phi_c(x_{i-}) \right.} - \frac{\partial \phi_t}{\partial \phi_c}_{\left| \phi_c(x_{i+}) \right.} \right) \Phi_s(\hat{x}_i) \right]$$

$$\frac{\partial \text{loss}}{\partial \theta_c} = \sum_{i \in [N_s]} 1(\Phi_s(\hat{x}_i)^T \Phi_t(x_{i+}) - \Phi_s(\hat{x}_i)^T \Phi_t(x_{i-}) < \alpha) \times$$

$$\left[ \frac{\partial \phi_t}{\partial \phi_c}_{\left| \phi_c(\hat{x}_i) \right.} \left| \phi_c(x_{i-}) \right. \frac{\partial \phi_c}{\phi_c(x_{i-})} \left| x_{i-} \right. \right] + \left[ \frac{\partial \phi_t}{\partial \phi_c}_{\left| \phi_c(\hat{x}_i) \right.} \left| \phi_c(x_{i+}) \right. \frac{\partial \phi_c}{\phi_c(x_{i+})} \left| x_{i+} \right. \right] \Phi_s(\hat{x}_i) \tag{2}$$