Supplement for Cross-Modal Knowledge Transfer Without Task-Relevant Source Data

Sk Miraj Ahmed[1], Suhas Lohit[2], Kuan-Chuan Peng[2], Michael J. Jones[2], and Amit K. Roy-Chowdhury[1]

[1] University of California, Riverside, CA 92507, USA
{sahme047@,amitrc@ece.}ucr.edu
[2] Mitsubishi Electric Research Laboratories (MERL), Cambridge, MA 02139, USA
https://www.merl.com {slohit,kpeng,mjones}@merl.com

1 Dataset example images

In Figure 1 and Figure 2 we show some example samples from SUN RGB-D dataset, whereas in Figure 3 and Figure 4, samples from RGB-NIR scene dataset has been shown. For both datasets, some random samples of Task-Relevant (TR) and and Task-Irrelevant (TI) classes are shown. As DIML dataset has most of the classes overlapped with SUN RGB-D, we do not show examples for that dataset here. For the TR classes, source data are discraded after training the source models and we transfer knowledge from those models to the unlabeled data of target modality. For the TI classes, we have paired samples from both modalities. Note that for all the cases, TR and TI classes are completely disjoint.

2 Calculation of pseudo-labels

For these steps, we mainly follow [1, 2]. We first compute the cluster centroids of all the classes, followed by linearly combining the centroids using the current learned weight vector. We then take each of the weighted features and label it according to it's nearest neighbors from the set of K weighted centroids. In the next step, we update the pseudo labels by repeating these steps. Below, we describe mathematically these steps in detail:

1. We first compute the cluster centroids of all the classes $k \in \{1, 2, \dots N\}$ induced by source $j \in \{1, 2, \dots, n\}$ for the 0-th iteration, by the following equation:

$$c_{kj}^{(0)} = \frac{\sum_{x_T \in \mathcal{X}_T^{m_T}} \left[\tilde{\mathcal{F}}_{S_j}^{m_S}(x_T)\right]_k \tilde{f}_j(x_T)}{\sum_{x_T \in \mathcal{X}_T^{m_T}} \left[\tilde{\mathcal{F}}_{S_j}^{m_S}(x_T)\right]_k} \tag{1}$$

where $\left[.\right]_k$ indicates the k-th element of the vector in argument , $\tilde{f}_j$ denotes the j-th source model's feature extractor and $\tilde{\mathcal{F}}_{S_j}^{m_S} = g_j \circ \tilde{f}_j$ represents the complete j-th source model from the last iteration.

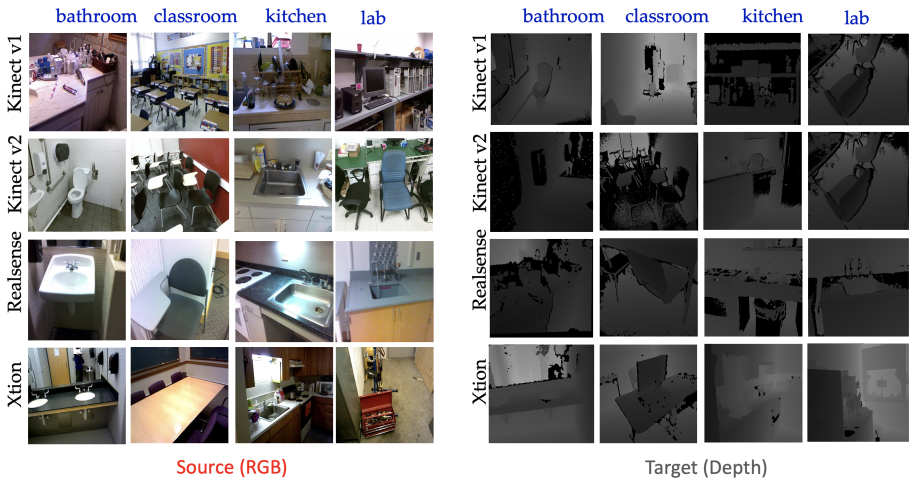

Fig. 1. SUN RGB-D TR samples. We show some example images of the four domains of SUN RGB-D. Both modalities from 4 out of 17 TR classes are shown here. We discard the RGB source data after training four source models and we do not use any label information for the target depth data.

Fig. 2. SUN RGB-D TI samples. We show some example images of the TI data from SUN RGB-D dataset. Six classes, each with paired example of RGB and depth are shown here. The TR and TI classes are completely disjoint.

2. In the next step, we linearly combine these centroids as well as the target features extracted from all the source models from last iteration, with the current learned weight vector ζ as follows:

$$c_k^{(0)} = \sum_{j=1}^{n} \zeta_j c_{k_j}^{(0)} \tag{2}$$

$$\bar{x}_T = \sum_{j=1}^{n} \zeta_j \tilde{f}_j(x_T) \tag{3}$$

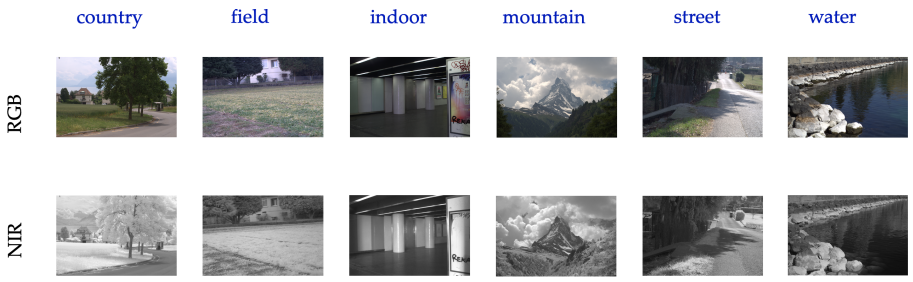

Fig. 3. RGB-NIR scene samples. We show some example images of the of RGB-NIR scene dataset. Both modalities of all 6 TR classes are shown here. We discard the source data after training the source model and we do not use any label information for the target data.

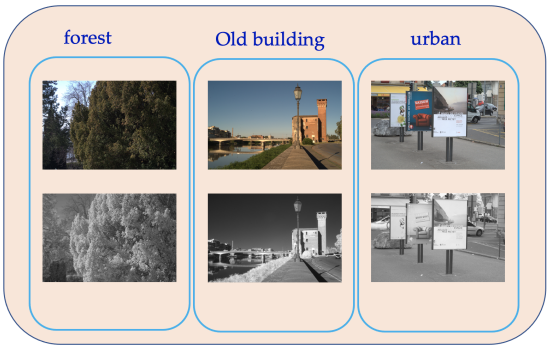

Fig. 4. RGB-NIR scene samples We show some example images of the TI data from RGB-NIR scene dataset. Three classes, each with paired example of RGB and NIR are shown here. The TR and TI classes are completely disjoint.

3. We take each of the weighted features and label it according to it's nearest neighbour from the set of K weighted centroids, *i.e.*, for a particular target feature, if the nearest neighbour is k-th centroid, we assign class label k for that particular feature. The assigned pseudo-label $\hat{y}_T^{i(0)}$ for the i-th target feature $\bar{x}_T^i$ at iteration 0 is calculated as:

$$\hat{y}_T^{i(0)} = \arg\min_k \|\bar{x}_T^i - c_k^{(0)}\|_2^2 \tag{4}$$

4. We update the pseudo-labels in the next iteration by repeating the steps as follows:

$$c_{k_j}^{(1)} = \frac{\sum_{x_T \in \mathcal{X}_T^{m_T}} \mathbf{1}\{\hat{y}_T^{(0)} = k\} \tilde{f}_j(x_T)}{\sum_{x_T \in \mathcal{X}_T^{m_T}} \mathbf{1}\{\hat{y}_T^{(0)} = k\}} \tag{5}$$

where, $\mathbf{1}\{.\}$ is an indicator function operator which takes value 1, when its argument is true.

$$c_k^{(1)} = \sum_{j=1}^{n} \zeta_j c_{k_j}^{(1)} \tag{6}$$

$$\hat{y}_T^{i(1)} = \arg\min_k \|\bar{x}_T^i - c_k^{(1)}\|_2^2 \tag{7}$$

Following the protocol of [1], we take $\hat{y}_T^{(1)}$ as the final pseudo-label $\hat{y}_T^i$, without further reiteration.

Finally the *pseudo-label cross entropy* loss $\mathcal{L}_{pl}$ is calculated as follows:

$$\mathcal{L}_{pl} = -\frac{1}{n_T} \sum_{i=1}^{n_T} \sum_{k=1}^{K} \mathbf{1}\{\hat{y}_T^i = k\} \log \left[\mathcal{F}_T^{m_T}(x_T^i)\right]_k. \tag{8}$$

3 More details about datasets

SUN RGB-D[3]: The 17 common scene classes shared among the four domains are *bathroom, classroom, computer room, conference room, corridor, discussion area, home office, idk, kitchen, lab, living room, office, office kitchen, printer room, reception room, rest space, study space.*

The 28 scene classes used as TI data are *basement, bedroom, book store, cafeteria, coffee room, dancing room, dinette, dining area, dining room, exhibition, furniture store, gym, home, study, hotel room, indoor balcony, study space, laundromat, lecture theatre, library, lobby, mail room, music room, office dining, play room, reception, recreation room, stairs, storage room.*

DIML RGB+D[4]: The 6 scene classes used as TR data are *bathroom, classroom, computer room, kitchen, corridor, living room.*

The 12 scene classes used as TI data are *bedroom, billiard hall, book store, cafe, church, hospital, laboratory, library, metting room, restaurant, store, warehouse.*

RGB-NIR Scene[5]: The 6 scene classes used as TR data are *country, field, indoor, mountain, street, water.*

The 3 scene classes used as TI data are *forest, old building, urban.*

4 Effect of regularization parameters

For the single source adaptation results, we empirically observe that, $(\lambda_{TI}, \lambda_d) = (0.5, 0.5), (0.5, 0.5), (0.1, 0.1), (0.5, 0.5)$ yields best result for Kinect v1, Kinect v2, Realsense and Xtion as targets respectively. For the DIML RGB+D datset, the parameters are set to be $(0.5, 0.5)$, whereas for the RGB-NIR scene dataset, it is set as $(0.01, 0.05)$. Note that, for all of the cases this hyper-parameters are chosen to balance the two loss terms. Our method always performs better than the baseline in the range of values of the hyperparameters we tested and are close to the best accuracies reported in the paper.

5 Network architectures

In our experiments, we take the Resnet50 [6] model pretrained on ImageNet as the backbone architecture for training the source models, the same way as [2, 7, 8]. Following the architectures used in [1, 9], we replace the last fully connected (FC) layer with a bottleneck layer containing 256 units, within which we add a Batch Normalization [10] (BN) layer at the end of the FC layer. A task specific FC layer with weight normalization [11] is added at the end of the bottleneck layer.

6 Training source models

For training the source models, we resize all the source images to 224×224. Moreover, to increase model robustness, we use smooth labels instead of one-hot encodings [12, 13] during this procedure. We set the maximum number of training epochs to 20 for all of the sources, irrespective of the datasets. We utilize stochastic gradient descent with a momentum 0.9 and weight decay 10^{-3}. The learning rates are set to 10^{-3} for the feature encoders (f_k's) and 10^{-2} for the added bottleneck layer. During adaptation and knowledge transfer to the target modality, a learning scheduler setting similar to [2, 9] $\theta = \theta_0(1+10p)^{-\frac{3}{4}}$ is used, where θ and θ_0 represent the current and initial learning rates and p is a real number between 0 to 1 which captures the training progress. θ_0 is set to be 10^{-3} for the feature encoders (f_k's) and 10^{-2} for the added bottleneck layers along with the source mixing weight parameters (ζ_k's). The maximum number of epochs during target adaptation is set to be 15.

7 Knowledge transfer details

During adaptation and knowledge transfer to the target modality, a learning scheduler setting similar to [2, 9] $\theta = \theta_0(1+10p)^{-\frac{3}{4}}$ is used, where θ and θ_0 represents the current and initial learning rates and p is real number between 0 to 1 which captures the training progress. θ_0 is set to be 10^{-3} for the feature encoders (f_k's) and 10^{-2} for the added bottleneck layers along with the source mixing weight parameters (ζ_k's). The learning rate decreases exponentially during the course of training The maximum number of epochs during target adaptation is set to be 15.

8 Modification of our algorithm in presence of TI *unpaired* data

In this section, we explore the scenario of inaccessibility of pairwise cross-modal data for TI classes. In practical scenario, one might not be able to acquire cross modal paired data. In this case we show that adversarial matching between two cross modal distributions works reasonably well. Inspired from [14], we propose

the following loss function in order to align the two cross modal data distributions which are unpaired. For this purpose, we incorporate a discriminator $\mathcal{D}$ in our framework.
Our adversarial loss has two components: (1) *True Discriminator loss* $\mathcal{L}_{TD}$ and (2) *Adversarial Discriminator loss* $\mathcal{L}_{AD}$. The first loss tries to distinguish between source and target, while the second loss is a proxy for the generator part of the well known usual adversarial loss component, which tries to fool the discriminator in such a way, so that it fails to distinguish between source and target domain. The generator is irrelevant in our framework since we are not generating any new samples, rather as a proxy of the generator we use the same discriminator as an adversary in the second loss. In short, the first loss tries to correctly classify the source and target samples, while the second loss tries to do the opposite. Now, we describe the losses mathematically below:

$$\mathcal{L}_{TD} = -\frac{1}{n_{TI}} \sum_{i=1}^{n_{TI}} \left[\log \mathcal{D}\Big(\sum_{j=1}^{n} \zeta_j \psi_j^i \Big) + \log \Big(1 - \mathcal{D}\Big(\sum_{j=1}^{n} \zeta_j f_j(x_{TI_i}^{m_T}) \Big) \Big) \right] \tag{9}$$

$$\mathcal{L}_{AD} = -\frac{1}{n_{TI}} \sum_{i=1}^{n_{TI}} \left[\log \mathcal{D}\Big(\sum_{j=1}^{n} \zeta_j f_j(x_{TI_i}^{m_T}) \Big) \right] \tag{10}$$

Note that, $\mathcal{L}_{TD}$ is essentially a cross entropy loss computed with source TI labels as 1 and target TI labels as 0, while $\mathcal{L}_{AD}$ is also a cross entropy loss but computed with target TI labels as 1. So, clearly $\mathcal{L}_{AD}$ will try to oppose the loss $\mathcal{L}_{TD}$, so that the source and target features are indistinguishable. So our overall adversarial loss $\mathcal{L}_{adv}$ is calculated as follows:

$$\mathcal{L}_{adv} = \mathcal{L}_{TD} + \lambda_{AD} \mathcal{L}_{AD} \tag{11}$$

where λ_{AD} is a regularization parameter to balance the two adversarial loss components. In the absence of TI paired data, the overall new objective function $\mathcal{L}_{tot}$ will be

$$\mathcal{L}_{tot} = \mathcal{L}_{ma} + \lambda_{adv} \mathcal{L}_{adv} + \lambda_d \mathcal{L}_d \tag{12}$$

To show the effectiveness of this loss, we conduct a small experiment in table 1. We transfer knowledge from the kv2 RGB model to unlabeled kv2 depth data. Due to time constraint we just run this algorithm with one random seed. λ_{AD} is set to be 10 to give slightly more importance to $\mathcal{L}_{AD}$ compare to $\mathcal{L}_{TD}$, since our ultimate goal is to learn a feature embedding that can not distinguish between source and target. Clearly we see that our new adversarial loss has an increment of almost 2.9% when used with $\mathcal{L}_{ma}$. Though this gain is not as high compare to the case of having paired TI data (see table 8 in main paper), it is still significant and has great potential. This result is intuitively expected and show that even if with unpaired TI data, we can reduce the modality gap in the absence of TR source data. We hypothesize that for the unpaired TI data case, it is possible to reach a certain extent of the level of performance when using paired TI data, by

Table 1. Effect of our proposed adversarial loss component. The accuracy column corresponds to single source adaptation from RGB to depth on *kv2* domain of SUN RGB-D dataset. We show the accuracy gain over using $\mathcal{L}_{ma}$ only inside the parentheses

$\mathcal{L}_{ma}$	$\mathcal{L}_{d}$	$\mathcal{L}_{adv}$	(a) accuracy (%)
✓			31.0
✓		✓	33.9 (↑2.9)
✓	✓	✓	**34.2** (↑3.2)

using relatively more amount of unpaired data. We will explore it in detail for the future work.

9 Future work, limitations and potential negative impact

Further studies are required to better understand the effect of amount of TI data and the diversity present in the data on the knowledge transfer results, which will require access to larger and more diverse datasets. Another interesting avenue for future direction is applying these ideas to other modalities like point clouds, medical imaging, etc.

The work in this paper is a general method for improving knowledge transfer from a source modality to a target modality with unlabeled data. The impact of this line of research is to make it easier to train networks for modalities and tasks where large amounts of data and labeled data are not available. This may lead to a wider deployment of deep learning for such modalities. For example in applications like person re-identification, one might have access to the source models trained on private IR labeled data, which they can use to adapt RGB unlabeled data using our method, in order to match people across cameras. Thus, these algorithms can of course be good or bad for society depending on the particular application in which these ideas are employed, the bias in the datasets being used etc. This is also in true in general for other source-free DA methods [1, 2]. Therefore, steps need to be taken to ensure positive and fair outcomes of this technology.

Bibliography

[1] Ahmed, S.M., Raychaudhuri, D.S., Paul, S., Oymak, S., Roy-Chowdhury, A.K.: Unsupervised multi-source domain adaptation without access to source data. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. (2021) 10103–10112

[2] Liang, J., Hu, D., Feng, J.: Do we really need to access the source data? source hypothesis transfer for unsupervised domain adaptation. In: International Conference on Machine Learning, PMLR (2020) 6028–6039

[3] Song, S., Lichtenberg, S.P., Xiao, J.: Sun rgb-d: A rgb-d scene understanding benchmark suite. In: Proceedings of the IEEE conference on computer vision and pattern recognition. (2015) 567–576

[4] Cho, J., Min, D., Kim, Y., Sohn, K.: Deep monocular depth estimation leveraging a large-scale outdoor stereo dataset. Expert Systems with Applications **178** (2021) 114877

[5] Brown, M., Süsstrunk, S.: Multi-spectral sift for scene category recognition. In: CVPR 2011, IEEE (2011) 177–184

[6] He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: Proceedings of the IEEE conference on computer vision and pattern recognition. (2016) 770–778

[7] Xu, R., Li, G., Yang, J., Lin, L.: Larger norm more transferable: An adaptive feature norm approach for unsupervised domain adaptation. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. (2019) 1426–1435

[8] Peng, X., Bai, Q., Xia, X., Huang, Z., Saenko, K., Wang, B.: Moment matching for multi-source domain adaptation. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. (2019) 1406–1415

[9] Ganin, Y., Lempitsky, V.: Unsupervised domain adaptation by backpropagation. In: International conference on machine learning, PMLR (2015) 1180–1189

[10] Ioffe, S., Szegedy, C.: Batch normalization: Accelerating deep network training by reducing internal covariate shift. In: International conference on machine learning, PMLR (2015) 448–456

[11] Salimans, T., Kingma, D.P.: Weight normalization: A simple reparameterization to accelerate training of deep neural networks. Advances in neural information processing systems **29** (2016) 901–909

[12] Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J., Wojna, Z.: Rethinking the inception architecture for computer vision. In: Proceedings of the IEEE conference on computer vision and pattern recognition. (2016) 2818–2826

[13] Müller, R., Kornblith, S., Hinton, G.: When does label smoothing help? arXiv preprint arXiv:1906.02629 (2019)

[14] Tzeng, E., Hoffman, J., Saenko, K., Darrell, T.: Adversarial discriminative domain adaptation. In: Proceedings of the IEEE conference on computer vision and pattern recognition. (2017) 7167–7176