Reply to Reviewer#1

1. How does the domain discriminator G_d work? G_d should be explained when it first appears.

Response:

Thanks for your comments and suggestions. G_d is a binary domain discriminator with all the source samples labelled as 0 and all the target samples labelled as 1.

G_d is one of the critical components in domain adversarial training [1, 2, 3]. The domain adversarial training aligns distributions by performing a minmax game: a domain discriminator is trained to separate the feature representation of the source domain from the one of the target domain, at the same time, feature generators are trained to deceive the domain discriminator. In this work, we adopt the training scheme proposed in [3] to enable the domain discriminator to identify samples of unknown class for open set domain adaptation. Then we can reduce the domain shift by optimizing:

$$\begin{aligned}\min_{G_f^t}\max_{G_d}\mathcal{L}(G_d, G_f^s, G_f^t) = \mathbb{E}_{x\sim p_t(x)}\left[\log\left(G_d\left(G_f^t(x)\right)\right)\right] \\ + \mathbb{E}_{x\sim p_s(x)}\left[\log\left(1 - G_d\left(G_f^s(x)\right)\right)\right].\end{aligned} \tag{R1.1}$$

Here $p_s(x)$ and $p_t(x)$ are probability distributions of source and target data respectively. G_f^s and G_f^t are the feature extractors for source and target samples, which share weights as in [3].

To clarify how the domain discriminator G_d and domain adversarial training works, we revised the manuscript by adding Section *3.2: Domain Adversarial Training* for better understanding.

2. In Eq.(2), p_s and p_t are probability distributions, so what do $p_s(z_t)$ and $p_t(z_t)$ mean? And why is the optimal G_d this?

Response:

Sorry for the misleading symbol notations. In Eq.(2) of the original manuscript, the subscript of z_t should be removed and z is the deep feature representation of a sample, *i.e.,* $z = G_f(x)$. As stated in *Section 3.1 open set domain adaptation* in the revised manuscript, $p_s(z)$ and $p_t(z)$ are probability distributions of feature representations in source and target domains respectively, as defined in [4]. In the revised manuscript, Eq.(2) is rewritten as:

$$G_d^*(z) = \frac{p_t(z)}{p_s(z) + p_t(z)}. \tag{R1.2}$$

Similar to [5], the proof for the property that Eq.(R1.2) gives the optimal G_d with fixed G_f^s and G_f^t is shown as follows,

Proof. For fixed G_f^s and G_f^t, we train G_d to maximize Eq.(R1.1):

$$\begin{aligned}\max_{G_d}\mathcal{L}(G_d, G_f^s, G_f^t) &= \int_x p_t(x)\log\left(G_d\left(G_f^t(x)\right)\right) + p_s(x)\log\left(1 - G_d\left(G_f^s(x)\right)\right)dx. \\ &= \int_z p_t(z)\log\left(G_d(z)\right) + p_s(z)\log\left(1 - G_d(z)\right)dz.\end{aligned} \tag{R1.3}$$

By taking the partial derivative of the objective Eq.(R1.3) with respect to G_d and exchanging the order of differentiation and integration with Leibnizs rule, the optimal G_d is in $[0, 1]$ given by Eq.(R1.2). □

The proof has been added in Section *3.2: Domain Adversarial Training* in the revised manuscript.

3. In Eq.(4) and Eq.(5), authors use the entropy of probabilities of known classes to measure the probability that the sample comes from a known class. Why do not directly use the output of G_c? The $K+1$ dimension of G_c (i.e. G_c^{K+1})) seems like the same probability.

Response:

Thanks for your suggestion. The presentation in the original manuscript may not be clear enough. As a matter of fact, we have directly used the output of G_c (G_c^{K+1}) to measure the probability that the sample comes from a known class. Such probability can be interpreted as the transferability for sample selection.

Due to lack of supervised signals to train a model to identify the unknown class, the output of G_d is used to approximate the probability that the sample comes from the unknown class, i.e., $G_c^{K+1} = G_d$. In the original manuscript, Eq.(3) measures the transferability (or probability of known class) by G_d as follows,

$$w_d(x) = 1 - G_d(z). \tag{R1.4}$$

To increase robustness, the transferability (or probability of known class) is also computed from a different perspective. Thus, the entropy of probabilities of known classes is utilized as a measurement as in Eq.(4) and Eq.(5), i.e.,

$$G_{c,\,known} = softmax([G_c^1, G_c^2, ..., G_c^K]). \tag{R1.5}$$

$$w_c(x) = 1 - H(G_{c,\,known}(z)). \tag{R1.6}$$

The intuition behind this approximation is that samples from the known classes can be categorized correctly by $G_{c,known}$. As *entropy* measures the degree to which the probability of the model is spread out over different possible states, it is used to measure the uncertainty of predictions. Due to the overlapping in the marginal distributions across domains, target samples from the known classes could be categorized correctly by the classifier $G_{c,known}$ trained on the source samples. This leads to low entropy (high transferability) for the samples from the known classes. For the samples from the unknown class, because they cannot be classified into one of the K known classes, the uncertainty of the prediction measured by the entropy is large.

In the revised manuscript, we have modified Section *4.2* to give a clearer presentation of the transferability criterion.

4. In Eq.(6), why is the probability of being a certain known class and the probability of being the unknown class independent?

Response:

Thanks for your comments. This sentence may not be precise enough and should be modified as "Eq.(R1.4) and Eq.(R1.6) measure the transferability from two independent perspectives".

As defined in Eq.(R1.4) and Eq.(R1.6), w_d and w_c are calculated by the domain discriminator G_d and the classifier $G_{c,\,known}$, respectively. The domain discriminator G_d is trained to differentiate the source domain from the target one, while the classifier $G_{c,\,known}$ is learnt to give a correct class label to each sample from the source or target domain. Since the probabilities of domain and class label are independent of each other, Eq.(R1.4) and Eq.(R1.6) can be considered as two independent measurement to compute the transferability.

In the revised manuscript, we have modified Section *4.2* to give a more precise explanation for Eq.(6).

5. In Fig.3, $G_f s$ for the source domain and target domain share weights. However, samples from target domain should be transferred, so are there some differences between these two G_fs? Or are there some extra structures after G_f for target domain?

Response:

Thanks for your precious comments. In this work, G_f^s and G_f^t denote the feature extractors for source and target samples, respectively. They share weights as in [3], so there is no difference between them. Also, there is no extra structure after G_f^t for the target domain. Such network design has been widely adopted in existing domain adaptation works [3, 6, 7].

To transfer knowledge between domains, the domain adversarial training is formulated as follows,

$$\begin{aligned}\min_{G_f^t}\max_{G_d}\mathcal{L}(G_d, G_f^s, G_f^t) = &\mathbb{E}_{x\sim p_t(x)}\left[\log\left(G_d\left(G_f^t(x)\right)\right)\right] \\ &+ \mathbb{E}_{x\sim p_s(x)}\left[\log\left(1 - G_d\left(G_f^s(x)\right)\right)\right].\end{aligned} \tag{R1.7}$$

By optimizing Eq.(R1.7) with fixed G_d, G_f^t is confined to generate features to confuse the domain discriminator for target samples. Therefore the distribution of the target domain can be aligned with the distribution of the source domain. As the distributions are aligned, the classifier G_c trained with the source label information can perform well for target samples, so the knowledge learned on the source domain can be transferred to the target domain.

6. The writing could be improved, especially for some confusing description. Some typos and grammar errors should be corrected, such as "a certain known classes".

Response:

Thanks for your precious comments. As suggested, the paper has been proofread and the typos and grammar errors have been corrected in the revised manuscript.

References

[1] Y. Ganin, E. Ustinova, H. Ajakan, P. Germain, H. Larochelle, F. Laviolette, M. Marchand, V. Lempitsky, Domain-adversarial training of neural networks, The Journal of Machine Learning Research 17 (1) (2016) 2096–2030.

[2] E. Tzeng, J. Hoffman, K. Saenko, T. Darrell, Adversarial discriminative domain adaptation, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2017, pp. 7167–7176.

[3] K. Saito, S. Yamamoto, Y. Ushiku, T. Harada, Open set domain adaptation by backpropagation, in: Proceedings of the European Conference on Computer Vision, 2018, pp. 153–168.

[4] J. Zhang, Z. Ding, W. Li, P. Ogunbona, Importance weighted adversarial nets for partial domain adaptation, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2018, pp. 8156–8164.

[5] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, Y. Bengio, Generative adversarial nets, in: Advances in Neural Information Processing Systems, 2014, pp. 2672–2680.

[6] Z. Cao, K. You, M. Long, J. Wang, Q. Yang, Learning to transfer examples for partial domain adaptation, arXiv preprint arXiv:1903.12230 (2019).

[7] K. You, M. Long, Z. Cao, J. Wang, M. I. Jordan, Universal domain adaptation, in: The IEEE Conference on Computer Vision and Pattern Recognition, 2019.