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 identifier with all the source samples labelled as 0 and all the target samples labelled as 1.

G_d is an essential component in domain adversarial training. In the domain adversarial training, a domain discriminator is trained to separate the feature representation of the source domain from the target domain, and a feature generator is trained to deceive the domain discriminator. Formally, the training procedure can be written as,

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

Here $p_s(x)$ and $p_t(x)$ are probability distributions of source and target data respectively. G_d aims to label source samples as 0 and target samples as 1. G_f^s and G_f^t are the feature extractors for source and target samples, which share weights as in [1].

To clarify how the domain discriminator G_d and domain adversarial training works, we revised the manuscript by adding *Section 3.2. Preliminary: 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* of 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 [2]. 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 [3], the proof for the property that Eq.(R1.2) gives the optimal G_d is as follows,

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

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

We take the partial differential of the objective Eq.(R1.3) with respect to G_d, and apply the Leibnizs rule to exchange the order of differentiation and integration to achieve optimal G_d in $[0, 1]$ at Eq.(R1.2). □

The proof has been added in *Section 3.2. Preliminary: Domain Adversarial Training* of 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:

For convenience, here we use p_k to denote the probability that the sample comes from a known class . As a matter of fact, we do use the output of G_c to measure p_k by evaluating $K+1$ dimensional output (i.e. G_c^{K+1}). Eq.(3) of the original manuscript indicates the measurement, which is as follows,

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

Here G_d denotes the last dimension of G_c (*i.e.,* $G_d = G_c^{K+1}$). The output of G_d is an approximation of the probability that sample from the unknown class, since the model does not converge during training and lack of supervised information to identify the unknown class. Therefore w_d is an approximation of p_s. In order to measure p_s from a different perspective and make it more robust, we also utilize the entropy of probabilities of known classes as a measurement as indicated in Eq.(4) and Eq.(5). These equations are respectively as follows,

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

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

The intuition behind this approximation is that target samples from the known classes can be categorized by $G_{c,known}$. As *entropy* measures the degree to which the probability of the model is spread out over different possible states, it can be used to measure the uncertainty of predictions. Due to the overlapping in the marginal distributions, the target samples from the known classes should be categorized by the classifier G_c that trained on the source samples, leading to low entropy. And for samples from the unknown class, because they cannot be aligned with a specific class from the source domain, the prediction tends to be uncertain over K known categories, leading to high entropy.

In the revised manuscript, we revise *Section 4.2* to make 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:

These probability can be considered independent are mainly contribute to two reasons:

- The probability of being a certain known class is independent from the probability of being a certain domain.
- The proposed model use the probability of being the target domain to approximate the probability of being the unknown class.

In the following, we give more detailed explanations.

In domain adaptation, the probability of being a certain known class is independent from the probability of being a certain domain. That is because the classifier that leverages domain-invariant features gives the probability of being a certain known class. The domain discriminator that leverages domain-relevant features gives the probability of from a certain domain. Since the classifier and domain discriminator work independently, and domain-invariant and domain-relevant can be well separated from each other [4], these probabilities can be considered as independent in domain adaptation tasks.

In open set domain adaptation, we can not directly give the probability of being the unknown class due to the lack of supervised information. Instead, since samples from the unknown class are usually untransferable and are prone to be labeled as the target sample, we use the probability of being the target domain (*i.e.,* the output of G_c^{K+1}) to approximate the probability of being the unknown class. Thus, the probability of being a certain known class and the probability of being the unknown class can be considered independent in our work.

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, only one G_f is used for the source domain and target domain, and there is no extra structures after G_f for target domain. Such network designing is widely adopted in many works [1, 5, 6], and is demonstrated to be an effective way to transfer knowledge in domain adversarial training.

Specifically, the domain adversarial training is as follows,

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

G_f^s and G_f^t are the feature extractors for source and target samples, which share weights as in [1]. In the training of Eq.(R1.7), G_f^t is confined to generate features to confuse domain discriminator for target samples. Therefore the distribution of the target domain can be aligned with the distribution of the source domain. As the distribution is aligned, the classifier G_c that 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 pointing out the mistake. As suggested, the paper has been proofread and the language errors have been corrected in the revised manuscript.

References

[1] 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.

[2] 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.

[3] 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.

[4] X. Peng, Z. Huang, X. Sun, K. Saenko, Domain agnostic learning with disentangled representations, arXiv preprint arXiv:1904.12347 (2019).

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

[6] 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.