Supplementary Material

Summary of the Applied Elementary Adaptation Function

Our work provides an optimal solution to combine *single-domain* DA functions SCDA (Zhang et al. 2021b,a) to address the hidden *multi-subdomain* adaptation problem. In this section, we give a summary of their method.

The *single-domain* DA method of a pre-trained model mainly consists of two steps. i) One first computes transformation functions of each feature relying on the optimal transport theory. ii) Then, one selects and adapts only features that contribute significantly to domain adaptations.

Numerical Feature Adaptation For numerical attributes, features are adapted using the so-called *increasing arrangement* in the optimal transport theory. Let $\mathbb{C}_{num}$ be a set of indexes of numerical features. $\forall k \in \mathbb{C}_{num}$, $\forall x \in \mathbb{X}^t$, the target domain numerical feature adaptation function is expressed as

$$\mathcal{G}_k(x) = (F_k^{s^{-1}} \circ F_k^t)(x^k)\,,$$

where x^k is the k-th numerical dimension of the input x, and F_k^s and F_k^t represent the cumulative distribution functions of the k-th dimension of source and target domains.

Categorical Feature Adaptation Categorical dimensions use a stochastic mapping function for each unique value of a category. Let $\mathbb{C}_{cate}$ be a set of indexes of categorical features, and $\mathbb{E}_k = \{e_1^k, \ldots, e_{n_k}^k\}$ a set of n_k unique values of the k-th categorical feature. $\forall k \in \mathbb{C}_{cate}$, $\forall l, r \in \{1, \ldots, n_k\}$, the stochastic target domain categorical feature adaptation function is expressed as

$$P(\mathcal{G}_k(e_l^k) = e_r^k) = \frac{R_{l,r}^k}{\sum_{j=1}^{n_k} R_{l,j}^k}\,,$$

where $\boldsymbol{R}^k \in {\mathbb{R}_+}^{n_k \times n_k}$ is an estimated optimal transport plan of the k-th categorical feature.

Feature Selection The feature selection process consists of identifying features that contribute the most to domain adaptations. Zhang et al. (2021a) propose to use pseudo-label techniques to evaluate performances of adapting individual features and keep only the adaptations that significantly increase prediction performances.

Proof of Equations (6) and (7)

The domain adaptation aims to align the joint distribution between source and target domains, that is,

$$P(\tilde{X}^t, \tilde{Y}^t) = P(X^s, Y^s)\,,$$

where $\tilde{X}^t \in \mathcal{X}$ and $\tilde{Y}^t \in \mathcal{Y}$ are respectively the target domain input and output variables after the optimal adaptation. For a known number of subdomains K, we have

$$P(h_t(X^t; K, \mathbb{X}^*, \boldsymbol{S}^*)) = P(\tilde{Y}^t = 1)\,,$$

by definition. Analogously, with a variable number of subdomains, we have

$$P(h_t^*(X^t; \boldsymbol{A}^*)) = P(\tilde{Y}^t = 1)\,.$$

The given pre-trained source domain predictive model is often well-estimated such that it is the optimal one in the source domain, that is, $h_s(x) = P(Y^s = 1 | X^s = x)$, Consequently, we have $P(h_s(X^s)) = P(Y^s = 1)$. We know that $P(\tilde{Y}^t = 1) = P(Y^s = 1)$ is a necessary condition of $P(\tilde{X}^t, \tilde{Y}^t) = P(X^s, Y^s)$. Therefore,

$$P(h_t(X^t; K, \mathbb{X}^*, \boldsymbol{S}^*)) = P(h_s(X^s))\,,$$

and

$$P(h_t^*(X^t; \boldsymbol{A}^*)) = P(h_s(X^s))\,,$$

are proved.

Experiments

General Setup Details

The *multi-subdomain* DA method DCTN (Xu et al. 2018) addresses the setting where hidden subdomain labels are given. Therefore, we rely on estimating $\mathbb{X}$ to get separations of subdomains and use such subdomain labels to train DCTN adaptation models. The method MultiDA (Mancini et al. 2018, 2019) can discover hidden subdomains; thus, we do not provide subdomain labels to this method.

Hyper-parameters In an unsupervised case, hyper-parameters are chosen based on the prediction performances of test data of the source domain. In contrast, in a weakly supervised case, hyper-parameters are chosen relying on test data of source domain and weakly-labeled target domain

data. We define the hyper-parameter searching set as $\mathbb{H} = \{0.001, 0.003, 0.005, 0.007, 0.01\}$. For DAN and DANN, we fix the learning rate to 0.005 and search the weighting parameter between classification error and domain alignment error among $\mathbb{H}$. For all other methods (MCD, MultiDA, DCTN), we explore the learning rate among $\mathbb{H}$. As for the FineTune case, we freeze all layers except the last one and set the learning rate to 0.0005 to fine-tune the layer to fit weakly-labeled target domain data.

Adaptation Performance

Kaggle Adaptation Tasks. We illustrate the variance of performances of the Kaggle datasets under different adaptation tasks from Figure 1 to Figure 5. Figure 1 and 2 are in a weakly supervised setting, and the others are in an unsupervised setting. The reported performances are from 10 repetitions with different random states. In each repetition, we select hyper-parameters as what we have introduced in the previous section. Note that our NN HSAV and LGB HSAV proposals have small variances in general, despite the absence of a re-training process on them. In contrast, *multi-subdomain* DA methods (MultiDA, DCTN) are generally more sensitive to random states, as they have a more considerable variance than *single-domain* DA methods. Nonetheless, the best performances obtained by *multi-subdomain* DA methods are better than some *single-domain* DA methods. For example, from Figure 2 to Figure 5, the best performance of MultiDA is better than that of DANN and MCD methods.

To further understand the sensitivity of adaptation methods, we fix the random state and train DCTN, MultiDA, and DANN using different hyper-parameters. The results are illustrated in Figure 12. It is clear that, compared to the *single-domain* DA method DANN, the *multi-subdomain* DA methods MultiDA and DCTN are more sensitive to the change of hyper-parameters; thus, they are more challenging to estimate. As recent works mainly tackle an adaptation problem of image data, they provide little inspiration on hyper-parameters of an adaptation problem of tabular data addressed by this paper.

Real Data Adaptation Tasks. Figure 6 to Figure 11 show variances of performances in the real fraud detection datasets of different periods. Note that, in this adaptation task, weakly labeled target data reduce largely variances of DCTN under different random states. Especially for the task G-3 to B, both MultiDA and DCTN have achieved impressive adaptation performances. In the unsupervised setting, although LGB HSAV does not improve significantly LGB Baseline (Figure 9 and 10), it has in general a lower variance. Nonetheless, NN HSAV performs well in such two tasks. In the adaptation task G-3 to B, MultiDA performances the best compared to other methods. However, our propositions are the best in terms of prediction performances and variances in average.

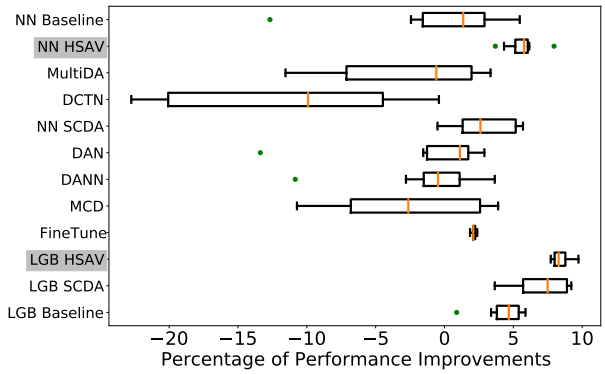

Figure 1: Variances of improvements in PR_AUC of the Kaggle adaptation task D-2 to M in a weakly supervised setting with different random states.

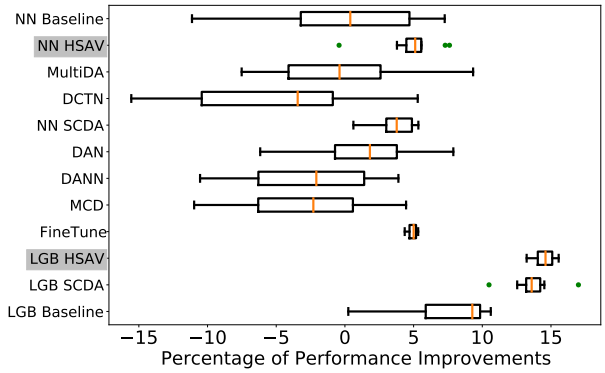

Figure 2: Variances of improvements in PR_AUC of the Kaggle adaptation task D-3 to M in a weakly supervised setting with different random states.

References

Mancini, M.; Porzi, L.; Bulo, S. R.; Caputo, B.; and Ricci, E. 2018. Boosting domain adaptation by discovering latent domains. In *CVPR*, 3771–3780.

Mancini, M.; Porzi, L.; Bulo, S. R.; Caputo, B.; and Ricci, E. 2019. Inferring latent domains for unsupervised deep domain adaptation. *IEEE TPAMI*.

Xu, R.; Chen, Z.; Zuo, W.; Yan, J.; and Lin, L. 2018. Deep cocktail network: Multi-source unsupervised domain adaptation with category shift. In *CVPR*, 3964–3973.

Zhang, L.; Germain, P.; Kessaci, Y.; and Biernacki, C. 2021a. Interpretable Domain Adaptation Using Unsupervised Feature Selection on Pre-trained Source Models.

Zhang, L.; Germain, P.; Kessaci, Y.; and Biernacki, C. 2021b. Target to Source Coordinate-Wise Adaptation of Pre-trained Models. In *ECML PKDD*, 378–394. Springer International Publishing.

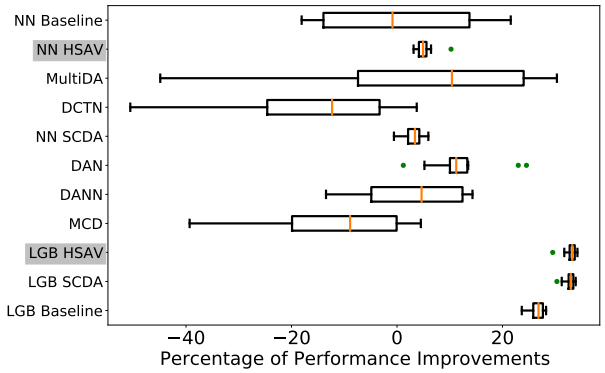

Figure 3: Variances of improvements in PR_AUC of the Kaggle adaptation task D-1 to M in an unsupervised setting with different random states.

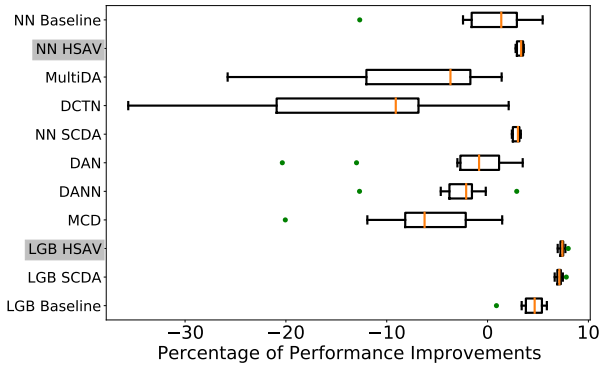

Figure 4: Variances of improvements in PR_AUC of the Kaggle adaptation task D-2 to M in an unsupervised setting with different random states.

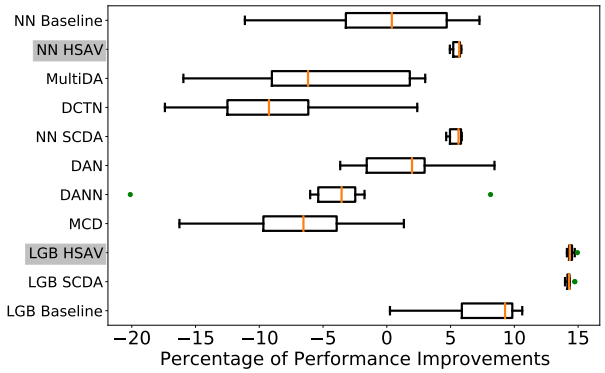

Figure 5: Variances of improvements in PR_AUC of the Kaggle adaptation task D-3 to M in an unsupervised setting with different random states.

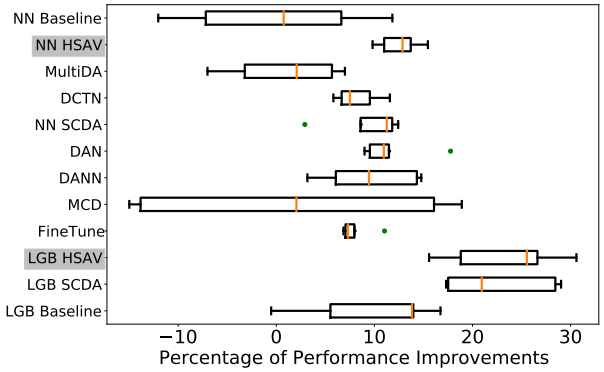

Figure 6: Variances of improvements in PR_AUC of the real data adaptation task G-1 to B in a weakly supervised setting with different random states.

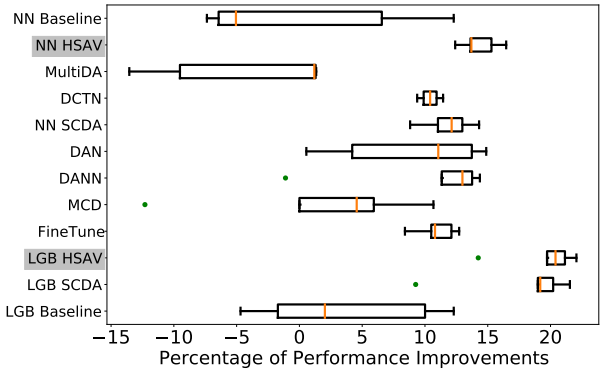

Figure 7: Variances of improvements in PR_AUC of the real data adaptation task G-2 to B in a weakly supervised setting with different random states.

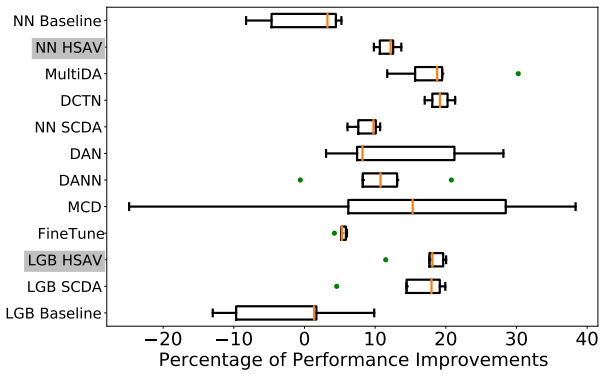

Figure 8: Variances of improvements in PR_AUC of the real data adaptation task G-3 to B in a weakly supervised setting with different random states.

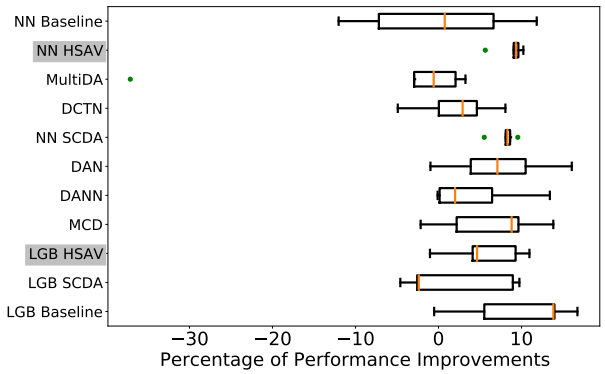

Figure 9: Variances of improvements in PR_AUC of the real data adaptation task G-1 to B in an unsupervised setting with different random states.

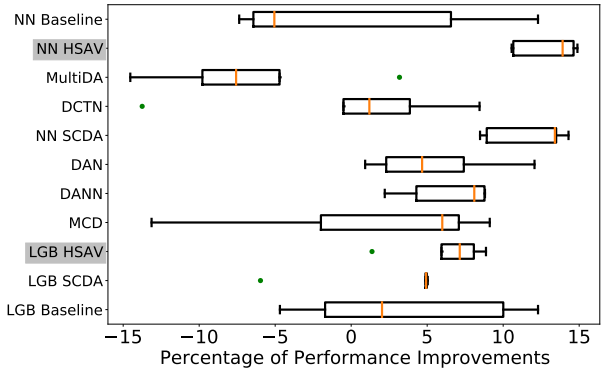

Figure 10: Variances of improvements in PR_AUC of the real data adaptation task G-2 to B in an unsupervised setting with different random states.

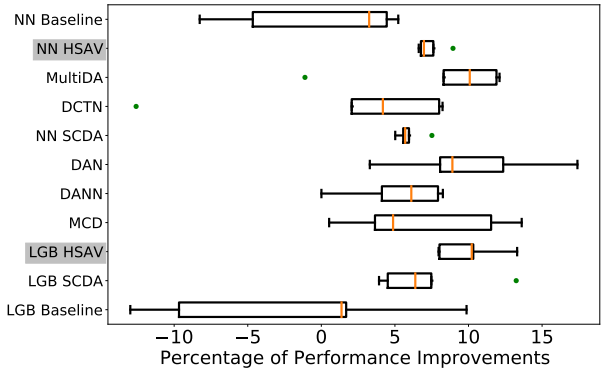

Figure 11: Variances of improvements in PR_AUC of the real data adaptation task G-3 to B in an unsupervised setting with different random states.

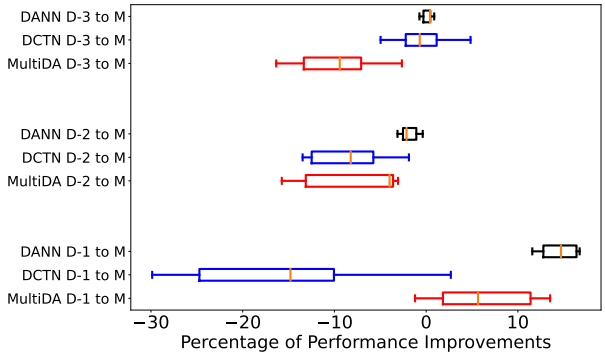

Figure 12: Variances of improvements in PR_AUC of Kaggle adaptation tasks in an unsupervised setting with different hyper-parameters.