Class-wise Feature Distribution Matching Regularization for Domain Generalization

Seungmin Lee
Seoul National University
profile2697@gmail.com

Abstract

Domain generalization (DG) aims to learn a model that generalizes well to an unseen domain (target domain), which has a different distribution than known domains (source domains). Many of the previous works try to learn domain-invariant features. These methods adopt a loss that tries to match the whole distributions of the source domains. However, these works are sub-optimal because they rarely utilize task-specific information, such as class labels. Concerning the information, we propose a simple regularizing method called Class-wise Feature Distribution Matching (FDM). The proposed method induces a model to produce similar features when the labels of examples are the same, regardless of the examples' domains. By doing this, the model is expected to learn more task-specific and invariant features than the previous works. To demonstrate the proposed methods, we conduct experiments on various settings. The proposed method consistently shows improvement compared to baseline. However, the improvement is marginal, and additional analysis reveals that domain-invariant features do not guarantee high performance.

1. Introduction

Deep learning has been remarkably successful in many areas [13, 17, 29, 23]. However, many studies find out deep learning methods are hard to generalize when they encounter an unseen domain, which has a different data distribution than domains used for training [19, 18, 19, 33, 9, 3, 24]. This problem called *domain shift* [34]. For alleviating the domain shift problem, many studies have been carried out on different assumptions. Domain Adaptation (DA) assumes there are two domains. The first is a fully-labeled source domain, and the other is a sparsely labeled or totally unlabeled target domain. Otherwise, domain generalization (DG) assumes there are some fully-labeled source domains, but the target domain is totally unavailable. DG is a challenge but important research area because generalization to other domains is crucial to make a safe artificial intelligence. Moreover, it is helpful to understand how deep neural networks see the world.

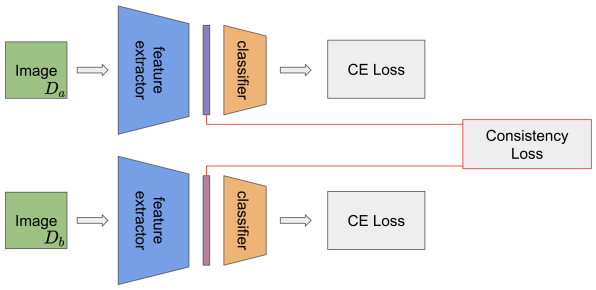

Figure 1: **Overview of the Class-wise Feature Distribution Matching Regularization** Assume examples that have the same class but different domains D_a and D_b. The proposed regularization makes the intermediate representation of the examples similar using consistency loss. We can expect the proposed consistency loss to make the feature extractor produce domain-invariant features when the examples have the same labels.

Existing DG studies can be classified into several categories depending on their strategies. Some methods proposed novel model architectures that are robust to domain shift [16, 18]. Others suggested learning algorithms aim to induce a model to fit in a more robust minimum [20, 19, 1]. The others adopted losses to learn domain-invariant features by matching feature distributions of the source domains [12, 25, 22]. Although these domain-invariant feature learning methods work well, the methods are sub-optimal because they do not utilize task-specific information such as class labels explicitly.

Our approach also aims to learn domain-invariant features, but the proposed method explicitly adopts the task-specific information. More specifically, we add a simple consistency loss that provokes the model to produce similar features when the labels are the same. By using this simple constraint, the model is expected to learn features that are

adequately semantic and invariant across domains. Moreover, the proposed method does not require any additional layers or components. Fig. 1 shows the overview of the proposed method.

To demonstrate the efficiency of the proposed method, we conducted experiments on various datasets [8, 18] and models [15]. In the tests, the proposed regularization method consistently improves the performance of the model compared to the baseline method. However, the gain is marginal, and further analysis shows that domain-invariant features do not guarantee reliable performance.

We can summarize our contributions in this works as follows: 1) we propose a simple regularization term that can make a network produce domain-invariant features without additional layers or components. 2) we conduct experiments on various settings to demonstrate the proposed method can improve the performance compared to baseline. 3) we provide a further analysis that indicates the domain-invariant features do not guarantee better performance, and the invariant in terms of distribution is less critical than the invariant in terms of performing the task at hand.

2. Related Works

2.1. Multi-Domain Learning and Multi-Source Domain Adaptation

The primary purpose of Multi-Domain Learning (*MDL*) is to learn a single model that can compactly represent all domains with a smaller number of parameters [35, 27, 2, 28]. For this purpose, Bilen *et al.* [2] adopts domain-specific parameters of instance normalization and batch normalization while using shared parameters in other layers. Rebuffi *et al.* [27] transforms the standard residual network architecture to share a significant amount of parameters between different domains.

Multi-Source Domain Adaptation (*MSDA*) also uses a set of source domains, but it additionally utilizes the images of an unlabeled target domain. The main focus of MSDA is to train a model that works well on the target domain without labels of it. Even though many studies have been conducted on single-source domain adaptation, there are a limited number of researches on MSDA [36, 5, 14, 26]. Chang *et al.* [5] proposes a domain-specific batch normalization with shared weights parameters and extends their method to MSDA. Peng *et al.* [26] suggests a way to reduce the moment distance between different source domains as well as reducing the distance between target and source domains. The moment distance measures the difference between feature distributions of two domains without concerning the task at hand.

MDL and MSDA are closely related to DG since DG also utilizes a set of domains as training data. However, DG is different from MDL in that the primary focus of DG is to learn semantic and domain-invariant features, not to learn compact representations. Furthermore, DG is more challenging than MSDA because the target domain is totally unavailable in DG.

2.2. Domain Generalization

Even though existing DG methods basically aim to learn domain-invariant features, the methods can be classified into several groups based on their approaches. The first group proposes a novel architecture [16, 18]. The methods separate domain-specific parameters and domain-agnostic parameters. After that, they only extract and utilize the domain-agnostic parameters for the unseen domain. The second group of methods suggests optimization algorithms that adopt episodic learning or self-supervised learning [20, 19, 1, 4]. For example, MLDG [19] constructs an episode by splitting the source domains into training domains and test domains in each iteration. The final group of methods uses losses that aims to learn domain-invariant features [12, 25, 22]. These methods often adopt maximum mean distribution (*MMD*) constraints. However, MMD simply tries to match the feature distributions of all available source domains without concerning the task at hand. Therefore, methods that adopted MMD can be sub-optimal [32, 31]. Otherwise, we propose a regularizing method using a simple consistency loss that explicitly utilizes the task-specific information. By using this simple loss, we expect that the model can learn domain-invariant but semantic features.

3. Proposed Method

3.1. Problem Setting and Notation

In DG, we assume that there are n source domains $\mathcal{D} = \{\mathcal{D}_1, \mathcal{D}_2, \dots, \mathcal{D}_n\}$ where $\mathcal{D}_i$ indicates i-th source domain which contains sample-label pairs $\{x_i^j, y_i^j\}$. Using the source domains, our goal is to train a model h that performs well on unseen target domain $\mathcal{D}_t$. We assume the model h consists of a feature extractor f and a classifier c.

3.2. Deep-All Method

The Deep-All Method is a simple but effective baseline. In this method, we just aggregate all examples of source domains and train the model using the aggregated samples. If we work on a classification task, we can use cross-entropy loss as follows:

$$L_{all} = \mathbb{E}_{\mathcal{D}_s \sim \mathcal{D}} \left[\mathbb{E}_{\mathbf{x}_i, y_i \sim \mathcal{D}_s} \left[\mathbf{y}_i{}^T \log h(\mathbf{x}_i) \right] \right] \quad (1)$$

where $\mathbf{y}_i$ is a one-hot vector representation of y_i. This method is easy to train, but it has shown comparable performance than other works.

Src.	Trgt.	D-MTAE [12]	DSN [3]	CrossGrad [33]	DICA [25]	DANN [10]	TF-CNN [18]	MetaReg [1]	MLDG [19]	Epi-FCR [20]	JiGen [4]	Deep-All	FDM
C,P,S	A	60.3	61.1	61.0	64.6	63.2	62.9	63.5	66.2	64.7	**67.6**	60.4	61.7
A,P,S	C	58.7	66.5	67.2	64.5	67.5	67.0	69.5	66.9	72.3	71.7	70.2	**73.1**
A,C,S	P	91.1	83.3	87.6	**91.8**	88.1	89.5	87.4	88.0	86.1	89.0	85.3	87.4
A,C,P	S	47.9	58.6	55.9	51.1	57.0	57.5	59.1	59.0	65.0	**65.2**	61.6	62.3
Ave.		64.5	67.4	67.9	68.0	69.0	69.2	69.9	70.0	72.0	**73.4**	69.4	71.1

Table 1: Cross-domain object classification results (accuracy. %) on PACS using AlexNet.

3.3. Class-wise Feature Distribution Matching Regularization

We propose a consistency loss called Class-wise Feature Distribution Matching Regularization (*FDM*), which tries to make a model generate similar features when the labels are the same. The FDM loss is calculated as follows: For each iteration, the feature extractor produces features of the source domains. After that, we calculate the averages of the features by classes. Lastly, the FDM loss is calculated as a consistency loss between the averages and the features that share the same label. The FDM loss is defined as follows:

$$L_{fdm} = \mathbb{E}_{\mathcal{D}_s \sim \mathcal{D}} \left[\mathbb{E}_{\mathbf{x}_i, y_i \sim \mathcal{D}_s} \left[d(f(\mathbf{x}_i), \mathbf{m}_{y_i}) \right] \right] \quad (2)$$

where d measures the difference between $\mathbf{x}_i$ and $\mathbf{m}_{y_i}$. we used the square of Euclidean distance. $\mathbf{m}_{y_i}$ is the average feature of class y_i. For simplicity, $\mathbf{m}_{y_i}$ is calculated for each batch. The L_{fdm} regularize the feature extractor f to extract similar features when the classes are the same.

Finally, overall loss function is defined as a weighted sum of L_{all} and L_{fdm} as follows:

$$L = L_{all} + \lambda L_{fdm} \quad (3)$$

where λ is a hyperparameter that controlls the magnitude between the two losses. We found that using linear ramp-up on λ is helpful to stable training. Thus, we updated λ at epoch t with the following schedule:

$$\lambda^{(t)} = min(1, \frac{t}{T_r}) * \lambda_{max} \quad (4)$$

where T_r is the ramp-up period, and λ_{max} is the maximum weight of the L_{fdm}.

4. Experiments

4.1. Brief Summary of Datasets

In this section, we give summaries of datasets used in our experiments. We use two different object classification datasets: **VLCS** [8] and **PACS** [18]. VLCS dataset consists of four famous datasets which are PASCAL VOC2007 (V) [7], Labelme (L) [30], Caltech 101 (C) [21], and SUN09 (S) [6]. Otherwise, PACS are composed of four datasets that have more significant visual domain gaps than those of VLCS. PACS consists of Photo (P), Art painting (A), Cartoon (C), and Sketch (S).

4.2. Settings

In this section, we evaluate the proposed method on standard DG benchmarks. For datasets, we use PACS [18] and VLCS [8], which are commonly used for demonstrating DG algorithms. For networks, we adopt Alexnet [17] and ResNet-18 [15]. We found the architectures of Alexnet [17] used in each paper are different. Thus, we clarify the used structure in this paper is the same as Li *et al.* [20]. In each network, we use convolution layers as the feature extractor and fully-connected layers as the classifier. For hyperparameters, we set λ_{max} to 0.1 and T_r to 40 for all experiments. We train a model for 100 epochs and use the batch size 512 for all settings. Following the experiments' protocol used in Li *et al.* [18], we evaluate the performance on a validation set during each epoch to get the best model, and we re-evaluate the performance on a test set using the model. We denote the proposed method as FDM in the results tables.

4.3. Results on PACS + Alexnet

The experimental results on PACS + Alexnet can be seen on Table 1. The proposed method shows a +1.7% improvement compared to the Deep-All baseline, which indicates the proposed method could be helpful to DG tasks. Furthermore, In this setting, the proposed method shows better performance than the majority of the previous methods. Interestingly, the Deep-All baseline also exhibits better performance than many previous works, as mentioned in Li *et al.* [20]. Importantly, the proposed method shows better performance than all methods trying to match feature distribution without concerning the task at hand [11, 10, 25]. The proposed method also shows the best performance when the target domain is 'cartoon.'

4.4. Results on VLCS + Alexnet

To demonstrate the proposed method works well on another dataset, We evaluate FDM on VLCS + Alexnet. The results are on Table 2. First, we could not reproduce the baseline of other works. Our baseline has almost 6% lower performance than the Deep-All from Li *et al.* [20]. Therefore, we emphasize that the performance of our method has a large disadvantage than other methods in advance. In this setting, FDM shows better performance than the baseline model. However, the gain is only +0.7%, which is quite

Src.	Trgt.	D-MTAE [12]	LRE-SVM [37]	CrossGrad [33]	DICA [25]	DANN [10]	MetaReg [1]	MLDG [19]	Epi-FCR [20]	Deep-All ([20])	Deep-All	FDM
L,C,S	V	63.9	60.6	65.5	63.7	66.4	65.0	67.7	67.1	65.4	59.7	59.3
V,C,S	L	60.1	59.7	60.0	58.2	64.0	60.2	61.3	64.3	60.6	54.1	54.3
V,L,S	C	89.1	88.1	92.0	79.7	92.6	92.3	94.4	94.1	93.1	87.7	88.0
V,L,C	S	61.3	54.9	64.7	61.0	63.6	64.2	65.9	65.9	65.8	61.4	64.0
Ave.		68.6	65.8	70.5	65.7	71.7	70.4	72.3	72.9	71.2	65.7	66.4

Table 2: Cross-domain object classification results (accuracy. %) on VLCS using AlexNet.

marginal. These results indicate the proposed regularization could be a hard constraint. We will discuss this problem at 5. Still, the proposed method shows better performance than [37, 25]. However, it shows weak performance than others. The disadvantage mentioned above seems to be a big cause, so it could not be fair comparisons.

4.5. Results on PACS + ResNet-18

We evaluate the proposed method on PACS + ResNet-18 to test whether FDM works well using different network architecture. The estimated results are on table 3. The proposed method consistently shows better performance than the Deep-All baseline. However, the gain is only +0.5%, which indicates the proposed method can be redundant when the network structure is strong enough. Moreover, the proposed method only gives better performance than CrossGrad [33]. Nevertheless, the proposed method shows the best performance when the target domain is 'photo.' Interestingly, DANN [10] designed to match the feature distribution without concerning the task at hand also shows better performance than FDM and shows comparable performance than state-of-the-art methods. It indicates if the network is strong enough, and there is a set of source domains, a simple feature distribution matching would be enough.

5. Discussion

5.1. Analysis: Hyperparameter Sensitivity

For a further understanding of this work, we conduct hyperparameter sensitivity experiments on PACS + Alexnet. As in the previous experiment, we fix the value of T_r at 40 for all tests. Instead, we experiment by changing the λ_{max} value by 0.1, from 0 to 1. Note that $\lambda_{max} = 0$ is equivalent to the Deep-All baseline. The results are plotted in Fig. 2. The x-axis represents λ_{max}, and the left y-axis and the blue line denote the average L2 distance between the mean of representations of all examples and each example. We match the scale of the L2 distances by dividing the corresponding λ_{max}s. The right y-axis and the orange line represent mean accuracy on target domains.

The proposed method only exceeds the baseline when the λ_{max} is one of $\{0.1, 0.2, 0.3\}$. However, the difference between those three configurations is marginal, which means FDM is quite robust to hyperparameter in a small range. Nevertheless, the accuracies rapidly decrease when

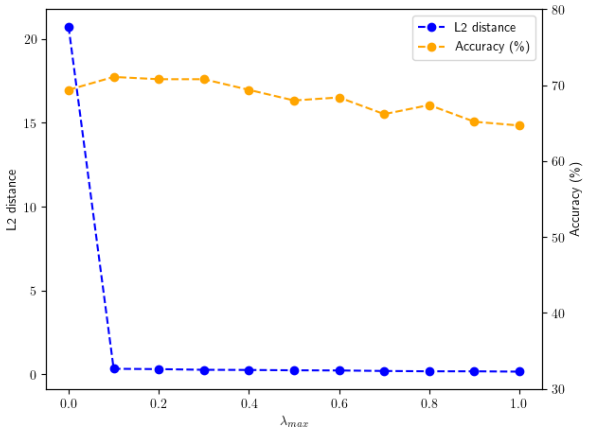

Figure 2: **The Plot of Hyperparameter Sensitivity on PACS + Alexnet** The orange line represents the mean accuracy of the four target domains. The blue line denotes the average L2 distance between the mean representation and the feature of each example.

we use larger λ_{max}. The L2 distances gradually decrease as we increase the value of λ_{max}, which is quite natural. However, when we compare them with the distance of the baseline, the difference is more than x100. It indicates the proposed method provokes the network to produce domain-invariant features. Interestingly, the difference in corresponding accuracies is quite marginal. Therefore, we could infer that the domain-invariant features do not guarantee high performance, and the proposed method may be a too hard constraint.

5.2. Why ResNet-18 Has a Smaller Gain Than Alexnet?

When we compare the performance gains of Alexnet (Table 1) and ResNet-18 (Table 3), we can see the performance gain in ResNet-18 is much smaller than that of Alexnet. It would indicate the redundancy of the proposed method. When we use the better architecture, the corresponding feature extractor would be able to extract discriminative features. **Those features may be variant in terms of distribution, but they could be considered invariant in terms of performing the task at hand.** Therefore, we need to design algorithms that can learn discriminative representations rather than trying to matching the feature dis-

Src.	Trgt.	CrossGrad [33]	DANN [10]	MetaReg [1]	MLDG [19]	Epi-FCR [20]	Deep-All	FDM
P,C,S	A	78.7	81.3	79.5	79.5	82.1	77.3	77.7
P,A,S	C	73.3	73.8	75.4	77.3	77.0	73.8	74.2
A,C,S	P	94.0	94.0	94.3	94.3	93.9	94.1	94.3
P,A,C	S	65.1	74.3	72.2	71.5	73.0	70.8	71.7
Ave.		77.8	80.8	80.4	80.7	81.5	79.0	79.5

Table 3: Cross-domain object classification results (accuracy. %) on PACS using ResNet-18.

tributions.

6. Future Works

In 5.2, we have discussed the difference between invariant in terms of feature distributions and invariant regarding performing the task at hand. Therefore, it would be better to design a new algorithm that can achieve task-specific invariant. It could be done by considering the relationship between feature distributions and decision boundaries. Therefore, adversarial attacks could be helpful because the technique is about the relationship.

7. Conclusion

We suggest a simple regularization technique called Class-wise Feature Distribution Matching Regularization. We conduct various experiments to demonstrate the efficiency of the proposed method. The proposed method consistently improves the performance of the baseline. However, further analysis reveals that the domain-invariant achieved by the regularizing technique is not critical to the performance. Moreover, through the study, we could learn the invariant in terms of distribution is less important than the invariant regarding performing the task at hand.

References

[1] Yogesh Balaji, Swami Sankaranarayanan, and Rama Chellappa. Metareg: Towards domain generalization using meta-regularization. In *NeurIPS*, 2018.

[2] Hakan Bilen and Andrea Vedaldi. Universal representations: The missing link between faces, text, planktons, and cat breeds. Technical report, 2017.

[3] Konstantinos Bousmalis, George Trigeorgis, Nathan Silberman, Dilip Krishnan, and Dumitru Erhan. Domain separation networks. In *NeurIPS*, 2016.

[4] Fabio Maria Carlucci, Antonio D'Innocente, Silvia Bucci, Barbara Caputo, and Tatiana Tommasi. Domain generalization by solving jigsaw puzzles. In *CVPR*, 2019.

[5] Woong-Gi Chang, Tackgeun You, Seonguk Seo, Suha Kwak, and Bohyung Han. Domain specific batch normalization for unsupervised domain adaptation. In *CVPR*, 2019.

[6] Myung Jin Choi, Joseph Lim, and Antonio Torralba. Exploiting hierarchical context on a large database of object categories. In *CVPR*, 2010.

[7] Mark Everingham, Luc Van Gool, Christopher K. I. Williams, John Winn, and Andrew. Zisserman. The pascal visual object classes (voc) challenge. *IJCV*, 2010.

[8] Chen Fang, Ye Xu, and Daniel N. Rockmore. Unbiased metric learning: On the utilization of multiple datasets and web images for softening bias. In *ICCV*, 2013.

[9] Yaroslav Ganin and Victor Lempitsky. Unsupervised domain adaptation by backpropagation. In *ICML*, 2015.

[10] Yaroslav Ganin, Evgeniya Ustinova, Hana Ajakan, Pascal Germain, Hugo Larochelle, François Laviolette, Mario Marchand, and Victor Lempitsky. Domain-adversarial training of neural networks. *JMLR*, 2016.

[11] Muhammad Ghifary, W Bastiaan Kleijn, Mengjie Zhang, and David Balduzzi. Domain generalization for object recognition with multi-task autoencoders. In *ICCV*, 2015.

[12] Muhammad Ghifary, W. Bastiaan Kleijn, Mengjie Zhang, and David Balduzzi. Domain generalization for object recognition with multi-task autoencoders. In *ICCV*, 2015.

[13] Ross Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *CVPR*, 2014.

[14] Jiang Guo, Darsh Shah, and Regina Barzilay. Multi-source domain adaptation with mixture of experts. In *EMNLP*, 2018.

[15] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, 2016.

[16] Aditya Khosla, Tinghui Zhou, Tomasz Malisiewicz, Alexei Efros, and Antonio Torralba. Undoing the damage of dataset bias. In *ECCV*, 2012.

[17] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. Imagenet classification with deep convolution neural networks. In *NeurIPS*, 2012.

[18] Da Li, Yongxin Yang, Yi-Zhe Song, and Timothy Hospedales. Deeper, broader and artier domain generalization. In *ICCV*, 2017.

[19] Da Li, Yongxin Yang, Yi-Zhe Song, and Timothy Hospedales. Learning to generalize: Meta-learning for domain generalization. In *AAAI*, 2018.

[20] Da Li, Jianshu Zhang, Yongxin Yang, Cong Liu, Yi-Zhe Song, and Timothy M. Hospedales. Episodic training for domain generalization, 2019.

[21] Fei-Fei Li, Fergus Rob, and Perona Pietro. Learning generative visual models from few training examples: an incremental bayesian approach tested on 101 object categories. In *CVPR Workshop on Generative-Model Based Vision*, 2004.

[22] Haoliang Li, Sinno Jialin Pan, Shiqi Wang, and Alex C Kot. Domain generalization with adversarial feature learning. In *CVPR*, 2018.

[23] Jonathan Long, Evan Shelhamer, and Trevor Darrell. Fully convolutional networks for semantic segmentation. *IEEE TPAMI*, 2017.

[24] Saeid Motiian, Marco Piccirilli, Donald A. Adjeroh, and Gianfranco Doretto. Unified deep supervised domain adaptation and generalization. In *ICCV*, 2017.

[25] Krikamol Muandet, David Balduzzi, and Bernhard Schölkopf. Domain generalization via invariant feature representation. In *ICML*, 2013.

[26] Xingchao Peng, Qinxun Bai, Xide Xia, Zijun Huang, Kate Saenko, and Bo Wang. Moment matching for multi-source domain adaptation. In *ICCV*, 2018.

[27] Sylvestre-Alvise Rebuffi, Hakan Bilen, and Andrea Vedaldi. Learning multiple visual domains with residual adapters. In *NeurIPS*, 2017.

[28] Sylvestre-Alvise Rebuffi, Hakan Bilen, and Andrea Vedaldi. Efficient parametrization of multi-domain deep neural networks. In *CVPR*, 2018.

[29] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster R-CNN: Towards real-time object detection with region proposal networks. *IEEE TPAMI*, 2017.

[30] Bryan C Russell, Antonio Torralba, Kevin P Murphy, and William T Freeman. Labelme: a database and web-based tool for image annotation. *IJCV*, 2007.

[31] Kuniaki Saito, Yoshitaka Ushiku, Tatsuya Harada, and Kate Saenko. Adversarial dropout regularization. In *ICLR*, 2018.

[32] Kuniaki Saito, Kohei Watanabe, Yoshitaka Ushiku, and Tatsuya Harada. Maximum classifier discrepancy for unsupervised domain adaptation. In *CVPR*, 2018.

[33] Shiv Shankar, Vihari Piratla, Soumen Chakrabarti, Siddhartha Chaudhuri, Preethi Jyothi, and Sunita Sarawagi. Generalizing across domains via cross-gradient training. In *ICLR*, 2018.

[34] Hidetoshi Shimodaira. Improving predictive inference under covariate shift by weighting the log-likelihood function. *JSPI*, 2000.

[35] Yongxin Yang and Timothy M. Hospedales. A unified perspective on multi-domain and multi-task learning. In *ICLR*, 2015.

[36] Han Zhao, Shanghang Zhang, Guanhang Wu, José M. F. Moura, Joao P Costeira, and Geoffrey J Gordon. Adversarial multiple source domain adaptation. In *NeurIPS*. 2018.

[37] Li Niu Zheng Xu, Wen Li and Dong Xu. Exploiting low-rank structure from latent domains for domain generalization. In *ECCV*, 2014.