TOWARDS DOMAIN GENERALIZATION IN UNDERWATER OBJECT DETECTION

Hong Liu[1,2], Pinhao Song[1,2], Runwei Ding[1,2]*

[1]Key Laboratory of Machine Perception, Shenzhen Graduate School, Peking University
[2]Peng Cheng Laboratory

ABSTRACT

A General Underwater Object Detector (GUOD) should perform well on most of underwater circumstances. However, with limited underwater dataset, conventional object detection methods suffer from domain shift severely. This paper aims to build a GUOD using small underwater dataset with limited types of water quality. First, we propose a data augmentation method Water Quality Transfer (WQT) to increase domain diversity of the original small dataset. Second, for mining the semantic information from data generated by WQT, Domain Generalization YOLO (DG-YOLO) is proposed, which consists of three parts: YOLOv3, Domain Invariant Module and Invariant Risk Minimization penalty. Finally, experiments on original and synthetic URPC2019 dataset prove that WQT combined with DG-YOLO achieves promising performance of domain generalization in underwater object detection. The source code can be found at https://github.com/mousecpn/DG-YOLO

Index Terms— domain generalization, object detection, underwater, data augmentation, domain invariance

1. INTRODUCTION

More and more research institutes and scientists consider attaching camera to underwater robots such as Autonomous Underwater Vehicles and Remotely Operated Vehicles in order to perform different underwater tasks, such as marine organism capturing, ecological surveillance and biodiversity monitoring. Underwater object detection is an indispensable technology to fulfill these tasks.

In application, once an underwater object detector aiming at certain categories has been trained, we hope this detector can be applied in any underwater circumstances. As a result, it is necessary to build a General Underwater Object Detector (GUOD). However, it faces three kinds of challenges:

(1) It is much harder to obtain underwater images, and the annotation task usually needs experts to accomplish, which is costly. Therefore, labeled dataset of underwater object detection is extremely limited, inevitably leading to overfitting of deep model. Data augmentation aims at solving the problem

of insufficient data. There are three types of augmentation. First, geometrical transformations (e.g., horizontal flipping, rotation, patch crop [1], and perspective simulation [2]) have been proved effective in various fields. Second, cut-Paste-based methods (e.g., randomly cut and paste [3], Mixup [4], CutMix [5], and PSIS [6]) help model learn contextual invariance. Third, domain-transfer based methods (e.g., SIN [7]) force model to focus more semantic information.

(2) The contradiction between speed and performance becomes even more critical. Real-time processing is a common requirement in robotics field. However, it is impractical to equip underwater robots with high performance hardware. Some works focus on the speed of deep learning model but keep good control of performance decrease, such as MobileNet [8], SSD [1], and YOLOv3 [9].

(3) Deep model severely suffers from domain shift, but a GUOD should be invariant of water quality, which can not only work well in oceans, but also in lakes and rivers. This can be seen as a kind of domain generalization problem that a model is trained on source domains but evaluated on an unseen domain. Some domain adaptation (DA) (e.g., style consistency [10], DA-Faster RCNN [11]) and domain generalization (DG) (e.g., JiGEN [12], MMD-AAE [13], and EPi-FCR [14]) technologies are proposed before. Nevertheless, most of DG works focus on object recognition and DA works can not directly transplant to DG task, so their effectiveness still needs to be proved in DG object detection task.

This work aims to use small dataset with limited domains to train a domain-invariant detector. First, for tackling challenge (1), we propose a new augmentation Water Quality Transfer to transfer an underwater image to different style, which can enlarge the dataset and increase domain diversity. Second, for tackling challenge (2) and (3), DG-YOLO is proposed, which combines the ideas of domain adversarial training and invariant risk minimization to further mine the semantic information of underwater image. Finally, a lot of experiments are conducted on Underwater Robot Picking Contest 2019 (URPC2019) dataset to prove the effectiveness of our method.

ICIP 2020

Fig. 1. Water Quality Transfer.

Fig. 2. The pipeline of WQT and DG-YOLO. *YOLO Back-Bone + YOLO Detector* is original YOLOv3. Domain label comes from WQT.
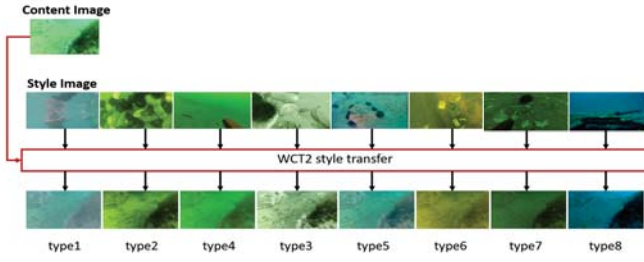
2. METHOD

2.1. Water Quality Transfer (WQT)

As Figure 1 shows, 8 images with different types of water quality are selected, and WCT^2 [15] is used to transfer UR-PC dataset to different types of water quality. The content image is from URPC's training set and validation set. In the following section, this seven types of training set are denoted as type1 to type7 and the corresponding validation set are denoted as *val_type1* to *val_type7*. As for type8, only the validation set is transferred to obtain *val_type8* without corresponding training set. Since our model will never train on type8 domain, *val_type8* is to test the domain generalization capacity of model.

2.2. Domain Generalization YOLO (DG-YOLO)

A review of YOLOv3. The real-time detector YOLOv3 [9] is a promising choice for underwater robots with limited calculation capacity. YOLOv3 is a one-stage object detector, using Darknet-53 as backbone. Compared with Faster R-CNN [16], YOLOv3 does not use region proposal network. It directly regresses the coordinates of bounding box and class information with a fully convolutional network. YOLOv3 divides an image into $S \times S$ cells, and each cell is responsible for the objects lie in the cell. The training losses of YOLOv3 consist of the loss of classification L_{cls}, the loss of coordination L_{coord}, loss of object L_{obj} and loss of no-object L_{noobj}:

$$L_{yolo} = L_{cls} + \lambda_{coord} \cdot L_{coord} + L_{obj} + \lambda_{noobj} \cdot L_{noobj}, \quad (1)$$

where λ_{coord} and λ_{noobj} are trade-off parameters.

Domain Invariant Module (DIM). Since DA and DG have some similarities, we modify the domain classifier proposed by [17] to apply in our DG task. Given a batch of input images as $\{X_1, X_2, ..., X_N\}$ from K different source domains, its corresponding domain labels are $\{d_1, d_2, ..., d_N\}$, in which N is the number of batch, $d_i \in \mathbb{R}^{K \times 1}$. Denoting that G is a feature extractor and D is a domain classifier, the domain loss L_d is defined as follows:

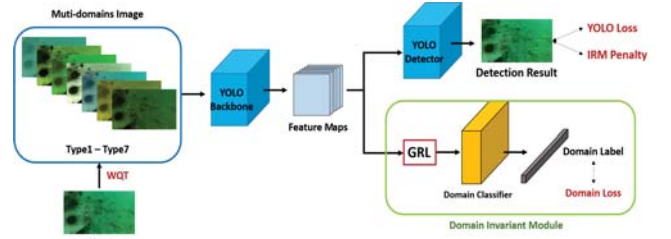$$L_d = \sum_i^N l_{CE}(D(G(X_i)), d_i), \quad (2)$$

where l_{CE} means categorical cross entropy. In application, domain label comes from WQT, and K is 7 corresponding to 7 types of water quality that WQT synthesizes. Domain loss for data from original dataset is not calculated.

IRM Penalty. Inspired by recent study [18], Invariant Risk Minimization (IRM) help learn an invariant predictor across multiple domain. Given a set of training environments (same meaning as domains) $e \in \mathcal{E}_{tr}$, our final goal is to achieve good performance across a large set of unseen but related environments $\mathcal{E}_{all}$ ($\mathcal{E}_{tr} \in \mathcal{E}_{all}$). However, directly using Empirical Risk Minimization (ERM) [19] will lead to overfitting on training environment and learn spurious correlation. In order to generalize well on unseen environments, IRM is a better choice to obtain invariance:

$$min_{\Phi:\mathcal{X} \sim \mathcal{Y}} \sum_{e \in \mathcal{E}_{tr}} R^e(\Phi) + \lambda \cdot ||\nabla_{r|r=1.0} R^e(r \cdot \Phi)||^2, \quad (3)$$

where Φ is the entire invariant predictors, $R^e(\Phi)$ is ERM term on environment e, $r = 1.0$ is a fixed scalar, $||\nabla_{r|r=1.0} R^e(r \cdot \Phi)||^2$ is invariance penalty, and $\lambda \in [0, \infty)$ is a trade-off parameter balancing the ERM term and the invariance penalty. To apply IRM in YOLOv3, the IRM penalty specially for YOLOv3 is designed as follows:

$$Pen_{coord} = \nabla_{r|r=1.0} || \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{I}_{i,j}^{obj} [(x_i - \hat{x}_i \cdot r)^2 + (y_i - \hat{y}_i \cdot r)^2,$$
$$+ (w_i - \hat{w}_i \cdot r)^2 + (h_i - \hat{h}_i \cdot r)^2] ||^2,$$

$$Pen_{cls} = \nabla_{r|r=1.0} || \sum_{i=0}^{S^2} \mathbb{I}_i^{obj} \sum_{c \in classes} l_{CE}(p_i(c), \sigma(a_i(c) \cdot r)) ||^2,$$

$$Pen_{obj} = \nabla_{r|r=1.0} || \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{I}_{i,j}^{obj} l_{CE}(C_i, \sigma(S_i \cdot r)) ||^2,$$

$$Pen_{noobj} = \nabla_{r|r=1.0} || \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{I}_{i,j}^{noobj} l_{CE}(C_i, \sigma(S_i \cdot r)) ||^2,$$

$$P_{IRM} = pen_{coord} + pen_{cls} + pen_{obj} + pen_{noobj}, \quad (4)$$

where $\mathbb{I}_i^{obj}$ denotes if object appears in cell i, $\mathbb{I}_{i,j}^{obj}$ denotes that the jth bounding box predictor in cell i is responsible for that

1972

prediction, σ is sigmoid operation, $\hat{a}_i(c)$ is the score of class c before sigmoid operation, $p_i \in \mathbb{R}^{K \times 1}$ is class label, $\hat{S}_i$ is the score of objects before sigmoid operation, $C_i \in \{0, 1\}$ is object label. $\{\hat{x}_i, \hat{y}_i, \hat{w}_i, \hat{h}_i\}$ is the bounding box outputted by YOLOv3, whose corresponding ground truth is $\{x_i, y_i, w_i, h_i\}$.

Penalty term is designed based on corresponding losses of YOLOv3. To be specific, r is added to corresponding places of losses. Square gradient of losses to r is the corresponding penalty term.

Network overview. An overview of our network is shown in Figure 2, we denote it as DG-YOLO. Compared to YOLOv3, DIM and IRM penalty are added. In details, the backbone of YOLOv3 (darknet-53) can be regarded as a feature extractor. The feature maps extracted from darknet will be fed into Gradient Reversal Layer (GRL) [17] first, which reverses the gradient when backpropagating for adversarial learning. After that, domain classifier distinguishes feature maps between domains. With the help of GRL and domain classifier, the backbone will be forced to abandon information of water quality to fool the domain classifier. As a result, DG-YOLO can make a prediction depending more on semantic information. Moreover, IRM penalty is calculated simultaneously with YOLO loss. Combining (1), (2) and (4) ,the total loss of DG-YOLO is:

$$L_{total} = L_{yolo} + \lambda_d \cdot L_d + \lambda_p \cdot P_{IRM}, \qquad (5)$$

λ_p and λ_d set to 1 in experiment. DG-YOLO has same inference speed as YOLOv3 since DIM and IRM penalty can be abandoned in inference stage. It should be emphasized that because domain label comes from WQT, DG-YOLO can not be used alone without WQT.

3. EXPERIMENTS AND DISCUSSIONS

3.1. Dataset

We evaluate WQT and DG-YOLO on a publicly available dataset: URPC2019[1], which consists of 3765 training samples and 942 validation samples over five categories: echinus, starfish, holothurian, scallop and waterweeds. Applying WQT on training set and validation set of URPC2019, we can synthesize *type1-7* for training and *val_type1-8* for validation. The performance on *val_type8* represents domain generalization capacity of model.

3.2. Training details

YOLOv3 and DG-YOLO are trained for 300 epochs and evaluated on original and all synthetic validation sets, with image resizing to 416×416. Models are trained on a Nvidia GTX 1080Ti GPU with PyTorch implementation, setting batch size to 8. Adam is adopted for optimization and learning rate sets

---

[1]www.cnurpc.org

to 0.001, with $\beta_1 = 0.9$ and $\beta_2 = 0.999$. IoU, confidence and non-max suppression threshold all set to 0.5. Accumulating gradient is leveraged, which is summing up the gradient and making one step of gradient descent in every two iterations. We do not use any other data augmentation on YOLOv3 and DG-YOLO unless we mention it.

3.3. Experiments of WQT

In this subsection, we analyze why WQT works. In Table 1, *Ori* means original URPC dataset, *baseline* means YOLOv3 is trained only on original dataset, and *ori+type1* means YOLOv3 is trained with original dataset and type1 dataset. *Full_WQT* means YOLOv3 is trained across type1 to type7. From Table 1, we can find three interesting points:

(1) Compared to *baseline*, it can be concluded that every group of augmentation improves the performance in original validation dataset. WQT can be used together with other data augmentation methods to obtain higher performance (last two rows of Table 1), which further proves its effectiveness. Besides, there is a phenomenon that WQT also helps model generalize better on other type of water quality in most of the cases. For example, *ori+type7* evaluates on *val_type3* get mAP 39.23%, 12.66% higher than *baseline*.

(2) We believe that there is a correlation between performance and similarity between water qualities. First, we use style loss proposed by [20] to represent style distance, and calculate the style distance between different types water quality. We feed style images type1 to type7 into WCT^2, and extract the feature maps at certain layers from both encoder and decoder, calculating style loss between any two types and obtaining H_{dist}. The result is shown in Table 2. Second, we take the data from column 3 to 9 (*val_+type1* to *val_type7*) and row 2 to 8 (model *ori+type1* to *ori+type7*) in Table 1, subtracting each row of this 7×7 matrix to the performance of corresponding type of *baseline*, getting H_{perf}. Using Pearson Correlation Coefficient and taking negative, it can be found that the correlation coefficient between style and performance is *0.4634*. From this analysis, it can be inferred that the increase of generalization capacity gaining from WQT is attributed to similarity between different types of water quality.

(3) To further analyze the findings of (2), model is evaluated on *val_type8* which is a very different style from type1 to type7. There is no doubt that the WQT-trained model will perform not only better on original dataset, but also across type1 to type7 dataset. However, the model still fails on *val_type8* (see Table 3). WQT is not enough for domain generalization.

3.4. Experiments of DG-YOLO

The effectiveness of DG-YOLO. WQT helps YOLOv3 to learn domain-invariant information, but the model still suffers from domain shift severely. In Table 3, it is shown that DG-YOLO further digs domain-invariant information from data,

1973

Table 1. The performance of model augmented by different types of water quality. Evaluation is performed on URPC2019 validation set of different types of water quality. All methods are implemented on YOLOv3.

Method	Evaluation (mAP)							
	val_ori	val_type1	val_type2	val_type3	val_type4	val_type5	val_type6	val_type7
baseline	56.45	18.72	16.83	26.57	10.71	23.66	9.38	29.04
ori+type1	56.66	52.39	27.66	42.07	14.25	42.79	20.96	41.07
ori+type2	56.71	18.90	51.86	39.44	24.89	34.51	6.21	45.85
ori+type3	57.78	18.01	29.96	53.10	15.63	35.07	5.68	41.20
ori+type4	58.33	16.80	33.50	41.57	53.85	35.52	3.72	42.30
ori+type5	57.63	35.35	30.73	42.04	20.04	53.12	19.41	42.28
ori+type6	57.19	21.64	35.63	42.19	24.37	36.04	51.22	46.15
ori+type7	58.43	7.57	34.81	39.23	15.52	32.77	3.88	52.36
Full_WQT	**58.56**	**55.93**	**53.60**	**57.48**	**54.95**	**56.08**	**53.51**	**54.29**
ori+rot+flip	62.53	14.81	18.29	31.36	8.89	24.95	5.34	33.18
Full_WQT+rot+flip	**63.83**	**60.57**	**57.71**	**60.38**	**58.96**	**59.84**	**58.43**	**60.53**

Table 2. Style distance between types of water quality.

	type1	type2	type3	type4	type5	type6	type7
type1	0	0.6281	0.1105	0.6893	0.0495	0.7239	0.6286
type2	0.6281	0	0.2860	0.0052	0.3311	0.0077	0.0033
type3	0.1105	0.2860	0	0.3435	0.0411	0.3575	0.2977
type4	0.6893	0.0052	0.3435	0	0.3747	0.0074	0.0037
type5	0.0495	0.3311	0.0411	0.3747	0	0.4024	0.3308
type6	0.7239	0.0077	0.3575	0.0074	0.4024	0	0.0094
type7	0.6286	0.0033	0.2977	0.0037	0.3308	0.0094	0

Fig. 3. SmoothGrad Visualization on "echinus" samples. Images are from original UPRC2019 dataset.

Table 3. Comparisons with other object detector and ablation study of DG-YOLO. IoU, NMS and Conf threshold all set to 0.5. The best checkpoints on *ori* and *val_type8* are selected respectively for different methods.

Method	ori	val_type8 (mAP)					
		echinus	starfish	holothurian	scallop	waterweeds	ave.
baseline (YOLOv3)	56.45	53.51	7.32	11.15	9.89	0	16.37
WQT-only	**58.56**	60.98	17.08	33.29	**39.02**	2.38	30.55
Faster-RCNN+FPN	58.20	29.49	5.91	9.13	1.07	10.40	11.21
SSD512	56.51	26.62	14.44	18.07	1.41	14.50	15.01
SSD300	50.66	27.31	14.57	13.62	3.01	2.98	12.31
WQT+DG-YOLO	54.81	**63.84**	**27.37**	**35.64**	36.88	**5.11**	**33.77**
WQT+DIM	58.06	58.78	18.55	26.64	21.82	4.39	26.03
WQT+P_{IRM}	57.01	54.99	25.98	32.90	29.25	0	30.63

obtaining 3.21% mAP improvement on *val_type8* compared to *WQT-only*. Besides, compared with other object detectors on *val_type8* performance, DG-YOLO shows its much better domain generalization capacity.

Ablation study. The result of ablation study is shown in Table 3. Compared to *WQT-only* on *val_type8*, *WQT+DIM* has 4.52% performance decrease and *WQT+P_{IRM}* has little improvement. However, *WQT+DG-YOLO* achieves 3.21% improvement, which suggests only by combining DIM and P_{IRM} can lead to better performance.

Visualization of DG-YOLO. It can not be ignored that there is performance decrease in original validation dataset of *WQT+DG-YOLO*. It is because *WQT-only* is "cheating", learning spurious correlation to make predictions. For example, waterweeds are green in greenish water, but they may turn
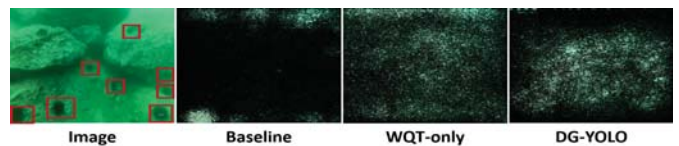
black in another type of water. Therefore, color of the object is not domain-invariant information, although it is convenient to use this spurious correlation to achieve good result in just one domain. The performance decrease of DG-YOLO can be interpreted that the model abandons the domain-related information and tries to learn domain invariant information from dataset. SmoothGrad [21] visualization technique is used to prove our hypothesis, finding the area that make model to believe there is echinus with probability higher than 95%. As is shown in Figure 3, *baseline* focuses on the shadow on the top left of image where there is no echinus. The pixel that *WQT-only* focuses is too dispersed, which means *WQT-only* learns spurious correlation. And the pixel DG-YOLO focuses is concentrated and exactly lie on the place where there is echinus. The visualization shows that DG-YOLO learns more semantic information than *baseline* and *WQT-only*.

4. CONCLUSION

This paper proposes a data augmentation method WQT and a novel model DG-YOLO to overcome three challenges a GUOD faces: limited data, real-time processing and domain shift. Leveraging WCT^2, WQT is intended to increase domain diversity of original dataset. With DIM and IRM penalty, DG-YOLO can further mine semantic information from dataset. Experiments on original and synthetic URPC2019 dataset prove remarkable domain generalization capacity of our method. However, since the performance of DG-YOLO in an unseen domain can still not reach similar level as that in the seen domains, there is still a lot to explore in this field.

5. REFERENCES

[1] Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, Cheng-Yang Fu, and Alexander C Berg, "Ssd: Single shot multibox detector," in *European conference on computer vision*. Springer, 2016, pp. 21–37.

[2] Hai Huang, Hao Zhou, Xu Yang, Lu Zhang, Lu Qi, and Ai-Yun Zang, "Faster r-cnn for marine organisms detection and recognition using data augmentation," *Neurocomputing*, vol. 337, pp. 372–384, 2019.

[3] Debidatta Dwibedi, Ishan Misra, and Martial Hebert, "Cut, paste and learn: Surprisingly easy synthesis for instance detection," in *Proceedings of the IEEE International Conference on Computer Vision*, 2017, pp. 1301–1310.

[4] Hongyi Zhang, Moustapha Cisse, Yann N Dauphin, and David Lopez-Paz, "mixup: Beyond empirical risk minimization," in *International Conference on Learning Representations*, 2018.

[5] Sangdoo Yun, Dongyoon Han, Seong Joon Oh, Sanghyuk Chun, Junsuk Choe, and Youngjoon Yoo, "Cutmix: Regularization strategy to train strong classifiers with localizable features," in *Proceedings of the IEEE International Conference on Computer Vision*, 2019, pp. 6023–6032.

[6] Wang Hao, Wang Qilong, Yang Fan, Zhang Weiqi, and Zuo Wangmeng, "Data augmentation for object detection via progressive and selective instance-switching," *arXiv preprint arXiv:1906.00358v2*, 2019.

[7] Robert Geirhos, Patricia Rubisch, Claudio Michaelis, Matthias Bethge, Felix A Wichmann, and Wieland Brendel, "Imagenet-trained cnns are biased towards texture; increasing shape bias improves accuracy and robustness," in *International Conference on Learning Representations*, 2018.

[8] Andrew G Howard, Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto, and Hartwig Adam, "Mobilenets: Efficient convolutional neural networks for mobile vision applications," *arXiv preprint arXiv:1704.04861*, 2017.

[9] Joseph Redmon and Ali Farhadi, "Yolov3: An incremental improvement," *arXiv preprint arXiv:1804.02767*, 2018.

[10] Adrian Lopez Rodriguez and Krystian Mikolajczyk, "Domain adaptation for object detection via style consistency," in *British Machine Vision Conference*, 2019.

[11] Yuhua Chen, Wen Li, Christos Sakaridis, Dengxin Dai, and Luc Van Gool, "Domain adaptive faster r-cnn for object detection in the wild," in *Computer Vision and Pattern Recognition (CVPR)*, 2018.

[12] Fabio M Carlucci, Antonio D'Innocente, Silvia Bucci, Barbara Caputo, and Tatiana Tommasi, "Domain generalization by solving jigsaw puzzles," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 2229–2238.

[13] Haoliang Li, Sinno Jialin Pan, Shiqi Wang, and Alex C Kot, "Domain generalization with adversarial feature learning," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 5400–5409.

[14] Da Li, Jianshu Zhang, Yongxin Yang, Cong Liu, Yi-Zhe Song, and Timothy M Hospedales, "Episodic training for domain generalization," in *International Conference on Computer Vision (ICCV)*, 2019.

[15] Jaejun Yoo, Youngjung Uh, Sanghyuk Chun, Byeongkyu Kang, and Jung-Woo Ha, "Photorealistic style transfer via wavelet transforms," in *International Conference on Computer Vision (ICCV)*, 2019.

[16] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun, "Faster r-cnn: Towards real-time object detection with region proposal networks," in *Advances in neural information processing systems*, 2015, pp. 91–99.

[17] Yaroslav Ganin and Victor Lempitsky, "Unsupervised domain adaptation by backpropagation," in *International Conference on Machine Learning*, 2015, pp. 1180–1189.

[18] Martin Arjovsky, Léon Bottou, Ishaan Gulrajani, and David Lopez-Paz, "Invariant risk minimization," *arXiv preprint arXiv:1907.02893*, 2019.

[19] Vladimir Vapnik, "Principles of risk minimization for learning theory," in *Advances in neural information processing systems*, 1992, pp. 831–838.

[20] Leon A Gatys, Alexander S Ecker, and Matthias Bethge, "Image style transfer using convolutional neural networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 2414–2423.

[21] Daniel Smilkov, Nikhil Thorat, Been Kim, Fernanda Viégas, and Martin Wattenberg, "Smoothgrad: removing noise by adding noise," *arXiv preprint arXiv:1706.03825*, 2017.