Supplementary Material : Min-Max Statistical Alignment for Transfer Learning

Samitha Herath[1,4], Mehrtash Harandi[2,4], Basura Fernando[1,5], and Richard Nock[1,3,4]
[1]The Australian National University, [2]Monash University, [3]The University of Sydney
[4]DATA61-CSIRO, Australia
[5]Human-Centric AI Programme, A*STAR, Singapore
Samitha.Herath@data61.csiro.au, Mehrtash.Harandi@monash.edu,
Fernando_Basura@scei.a-star.edu.sg, Richard.Nock@data61.csiro.au

We start this supplementary by providing the proof of **Theorem 1** from the main text. We repeat the theorem below for the convenience of the reader.

Theorem 1. *If the confusion function, g is a linear invertible transformation, $\boldsymbol{Q}$ with $\boldsymbol{Q}\boldsymbol{Q}^{-1} = \boldsymbol{Q}^{-1}\boldsymbol{Q} = \boldsymbol{I} \in \mathbb{R}^{d\times d}$ then the proposed min-max statistical alignment by confusion is equivalent to statistical alignment by minimization. Here, d is the dimensionality of both the domain input features.*

Proof. Let, $\boldsymbol{\Sigma}_0$ and $\boldsymbol{\Sigma}_1$ be the covariance matrices for the domains $\mathcal{D}_0$ and $\mathcal{D}_1$,respectively. Similarly, consider the domain means to be $\boldsymbol{\mu}_0$ and $\boldsymbol{\mu}_1$. Furthermore, let the feature confusion function, g be a linear transformation given by a square invertible matrix, $\boldsymbol{Q} \in \mathbb{R}^{d\times d}$. Hence, the confused feature means $\tilde{\boldsymbol{\mu}}_0$ and $\tilde{\boldsymbol{\mu}}_1$ for the two domains can be written as,

$$\tilde{\boldsymbol{\mu}}_0 = \boldsymbol{Q}^T\boldsymbol{\mu}_0, \text{and } \tilde{\boldsymbol{\mu}}_1 = \boldsymbol{Q}^T\boldsymbol{\mu}_1 \tag{1}$$

Furthermore, the confused feature covariances $\tilde{\boldsymbol{\Sigma}}_0$ and $\tilde{\boldsymbol{\Sigma}}_1$ are given by,

$$\tilde{\boldsymbol{\Sigma}}_0 = \boldsymbol{Q}^T\boldsymbol{\Sigma}_0\boldsymbol{Q}, \text{ and } \tilde{\boldsymbol{\Sigma}}_1 = \boldsymbol{Q}^T\boldsymbol{\Sigma}_1\boldsymbol{Q}. \tag{2}$$

From this, the confused feature KL divergence , $D_{KL}(\tilde{P}_0\|\tilde{P}_0)$ can be computed as,

$$\begin{aligned}
&D_{KL}(\tilde{P}_0\|\tilde{P}_0) = \\
&\quad \frac{1}{2}\Big(\text{tr}\big(\tilde{\boldsymbol{\Sigma}}_1^{-1}\tilde{\boldsymbol{\Sigma}}_0\big) + \big(\tilde{\boldsymbol{\mu}}_1 - \tilde{\boldsymbol{\mu}}_0\big)^T\tilde{\boldsymbol{\Sigma}}_1^{-1}\big(\tilde{\boldsymbol{\mu}}_1 - \tilde{\boldsymbol{\mu}}_0\big) \\
&\quad + \log\left(\frac{\det\tilde{\boldsymbol{\Sigma}}_1}{\det\tilde{\boldsymbol{\Sigma}}_0}\right) - d\Big) \\
&= \frac{1}{2}\Big(\text{tr}\big((\boldsymbol{Q}^T\boldsymbol{\Sigma}_1\boldsymbol{Q})^{-1}\boldsymbol{Q}^T\boldsymbol{\Sigma}_0\boldsymbol{Q}\big) \\
&\quad + \big(\boldsymbol{Q}^T\tilde{\boldsymbol{\mu}}_1 - \boldsymbol{Q}^T\tilde{\boldsymbol{\mu}}_0\big)^T\big(\boldsymbol{Q}^T\tilde{\boldsymbol{\Sigma}}_1\boldsymbol{Q}\big)^{-1}\big(\boldsymbol{Q}^T\tilde{\boldsymbol{\mu}}_1 - \boldsymbol{Q}^T\tilde{\boldsymbol{\mu}}_0\big) \\
&\quad + \log\left(\frac{\det\big(\boldsymbol{Q}^T\boldsymbol{\Sigma}_1\boldsymbol{Q}\big)}{\det\big(\boldsymbol{Q}^T\boldsymbol{\Sigma}_0\boldsymbol{Q}\big)}\right) - d\Big) \\
&= \frac{1}{2}\Big(\text{tr}\big(\boldsymbol{Q}^{-1}\boldsymbol{\Sigma}_1^{-1}\boldsymbol{Q}^{-T}\boldsymbol{Q}^T\boldsymbol{\Sigma}_0\boldsymbol{Q}\big) \\
&\quad + \big(\tilde{\boldsymbol{\mu}}_1 - \tilde{\boldsymbol{\mu}}_0\big)^T\boldsymbol{Q}\boldsymbol{Q}^{-1}\tilde{\boldsymbol{\Sigma}}_1^{-1}\boldsymbol{Q}^{-T}\boldsymbol{Q}^T\big(\tilde{\boldsymbol{\mu}}_1 - \tilde{\boldsymbol{\mu}}_0\big) \\
&\quad + \log\left(\frac{\det\big(\boldsymbol{Q}^T\big)\det\big(\boldsymbol{\Sigma}_1\big)\det\big(\boldsymbol{Q}\big)}{\det\big(\boldsymbol{Q}^T\big)\det\big(\boldsymbol{\Sigma}_0\big)\det\big(\boldsymbol{Q}\big)}\right) - d\Big) \\
&= \frac{1}{2}\Big(\text{tr}\big(\boldsymbol{\Sigma}_1^{-1}\boldsymbol{\Sigma}_0\big) + \big(\boldsymbol{\mu}_1 - \boldsymbol{\mu}_0\big)^T\boldsymbol{\Sigma}_1^{-1}\big(\boldsymbol{\mu}_1 - \boldsymbol{\mu}_0\big) \\
&\quad + \log\left(\frac{\det\big(\boldsymbol{\Sigma}_1\big)}{\det\big(\boldsymbol{\Sigma}_0\big)}\right) - d\Big) \\
&= D_{KL}(P_0\|P_0)
\end{aligned} \tag{3}$$

■

1. Network structures

Here we provide details of our network models. For our UDA experiments on Office31 dataset, we use the pretrained AlexNet [7] on ImageNet [1] and fine-tuned it accordingly. To obtain stable covariance matrices, we used an additional dimensionality reduction layer between $fc7$ and $fc8$ layers of the AlexNet where the dimensionality is reduced to 256. A similar modification to the AlexNet is used in prior work[2] for Office31 experiments. Table 1 provides details of the structure of the network used in UDA experiments on MNIST, SVHN, SYN. DIGITS, SYN. SIGNS, GTSRB, STL and CIFAR datasets. In Table 2 the details of the network models (f_1 and g) used in our toy data experiment (*see* Fig.1 in the main paper) are listed. Note that the feature extractor f_0 is an identity mapping for this experiment. Lastly, in Table 3, we report the details of the deep network models used in ZSL experiments.

Feature extractor (f_s, f_t)	
1	32 x 32 x 3 Image
2	Instance Normalization
3	3 x 3 conv, 64 (96) filters, leaky-ReLU
4	3 x 3 conv, 64 (96) filters, leaky-ReLU
5	3 x 3 conv, 64 (96) filters, leaky-ReLU
6	2 x 2 max-pool, stride 2
7	dropout, keep prob. = 0.5
8	Gaussian noise
9	3 x 3 conv, 64 (192) filters, leaky-ReLU
10	3 x 3 conv, 64 (192) filters, leaky-ReLU
11	3 x 3 conv, 64 (192) filters, leaky-ReLU
12	2 x 2 max-pool, stride 2
13	dropout, keep prob. = 0.5
14	Gaussian noise
Classifier Network (h)	
1	Input features
2	3 x 3 conv, 64 (192) filters, leaky-ReLU
3	3 x 3 conv, 64 (192) filters, leaky-ReLU
4	3 x 3 conv, 64 (192) filters, leaky-ReLU
5	global average pooling
6	10 (9) - fully connected
Confusion Network (g) - Residual Model	
1	Input features
2	3 x 3 conv, 64 (192) filters
3	leaky-ReLU
4	global average pooling

Table 1. The network structures for the UDA experiments on MNSIT, SVHN, SYN. DIGITS, SYN. SIGNS, GTSRB, STL and CIFAR. The values within parenthesis are used for the domain transformations STL↔CIFAR. All leaky-ReLU layers use 0.1 as the negative scale factor. The additive Gaussian noise layers use zero mean and a standard deviation of 0.0001. We use Batch-normalization for all convolutional layers. As in Shu *et al.* [8], we use Instance Normalization [10] at the input. Note that unlike the Office31 experiments, the outputs of f_s and f_t are 8×8 feature maps. Therefore, we use a convolutional network as the confusion network, g.

Feature extractor (f_1)	
1	2 - dimensional input features
2	2 - fully connected, Tanh
3	2 - fully connected, Tanh
Confusion Network (g)	
1	2 - dimensional input features
2	16 - fully connected, leaky - ReLU

Table 2. The network structures for the toy data experiment. Note that f_0 is an identity mapping. All leaky-ReLU layers use 0.1 as the negative scale factor.

Feature extractor (f_s)	
1	2048 dimensional features
2	512 (1024) - fully connected, leaky-ReLU
Feature Generator Network ($f_{att.}$)	
1	class attribute vectors
2	512 (1024) - fully connected, leaky-ReLU
3	dropout, keep prob. = 0.5
4	Gaussian noise
5	512 - fully connected, leaky-ReLU
6	dropout, keep prob. = 0.5
7	Gaussian noise
8	512 (1024) - fully connected, leaky-ReLU
9	dropout, keep prob. = 0.5
10	Gaussian noise
11	512 (1024) - fully connected, leaky-ReLU
Feature Classifier Network (h)	
1	512 (1024) dimensional features
2	number of classes - fully connected
Confusion Network (g) - Residual Model	
1	512 (1024) dimensional features
2	512 (1024) - fully connected, leaky-ReLU

Table 3. The network structures for the ZSL experiments. The values within parenthesis are used for the Sun dataset. All leaky-ReLU layers use 0.1 as the negative scale factor. The additive Gaussian noise layers use zero mean and a standard deviation of 1.0. We use Batch-normalization for all layers of the generator network (*i.e.*, $f_{att.}$).

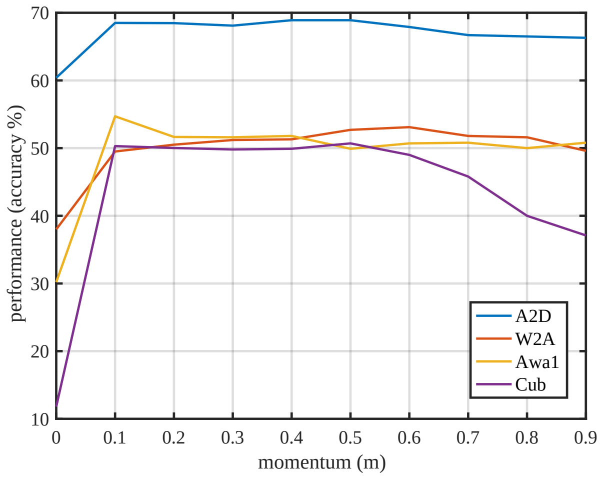
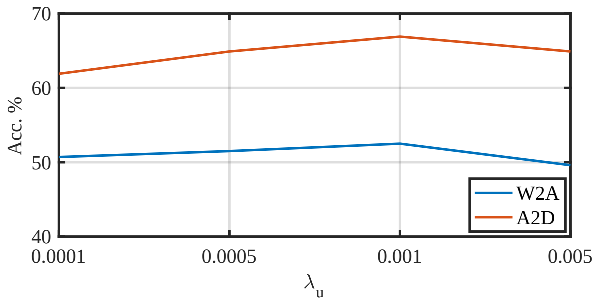


Figure 1. The robustness of the proposed min-max solution with respect to variations of the statistical accumulation momentum, m.



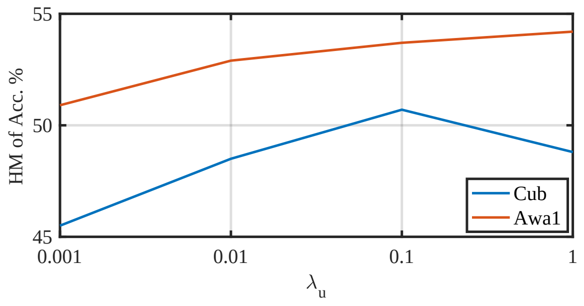


Figure 2. The robustness of the proposed min-max solution for the λ_u parameter in the loss function (*see* equations 8 and 11 in the main paper) for two UDA sets (Left) and two ZSL sets (Right).

2. Moment accumulation

In our experiments, we used statistical accumulation with $m = 0.5$ for UDA and $m = 0.1$ for ZSL. In Fig. 1, we report the robustness of the proposed min-max solution. For this experiment, we select two UDA experiment sets and two ZSL experiment sets. We observe that in all cases, moment accumulation is beneficial for statistical alignment. Furthermore, the performance of the solutions stays consistent for a wide range of momentum values (*i.e.*, $0.1 \leq m \leq 0.6$). We observe that in the case of ZSL, the moment accumulation is essential. We conjecture that this is due to the stability accumulation can bring in to the min-max learning.

3. Statistical loss weight λ_u

In Fig. 2, we report effects of changing the statistical loss weight, *i.e.*, λ_u in Eq. 8 and Eq. 11 of the main text. For this experiment, we select two UDA sets with $\lambda_u \in \{0.0001, 0.0005, 0.001, 0.005\}$ and two ZSL experiment sets with $\lambda_u \in \{0.001, 0.01, 0.1, 1\}$. In both experiments, consistent performance can be attained in a wide range of values for λ_u, suggesting robustness and easy-tuning. We observe that relatively smaller values of λ_u are useful for the UDA sets in comparison to the ZSL sets.

4. Mini-batch creation for GZSL

We train the final classifier (*see* § 3.2.1 of the main paper) with mini-batches containing a mixture of real seen classes instances and generated unseen class instances. To improve our classifier's robustness to unseen classes, we include generated

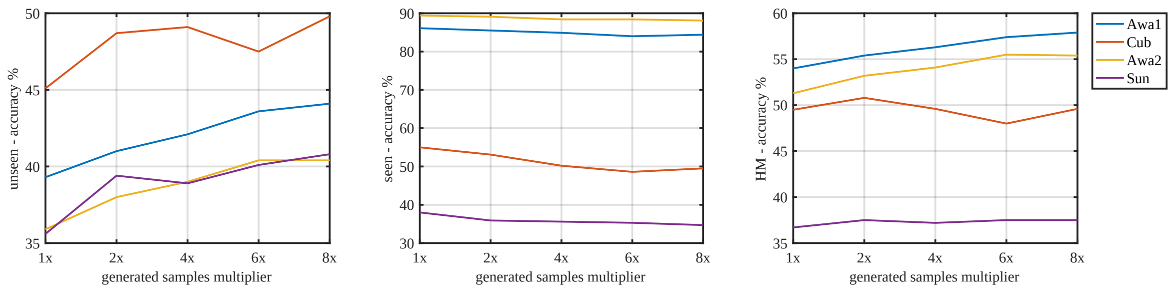


Figure 3. The performance of the proposed ZSL solution with different proportions of generated class instances.

class instances as 2x the number of real instance for each mini-batch in our reported experiments (*see* Table 3 in the main paper). In Fig. 3, we report the performance of our solution when the number of unseen class instances are increased by 2x, 4x, 6x and 8x. Here, we observe a consistent improvement in performance, except for Cubs experiment, by increasing the number of generated samples in mini-batches. Furthermore, we observe that the unseen class performance can be significantly improved by increasing the proportion of generated unseen class training samples.

5. Confusion network design

In all our reported experiments, we used confusion networks with a residual skip connection. In this section we provide further insights into our proposed confusion network design by comparing the performance of several network structures (*see* Fig. 4). In Table 4 we report the performance of different confusion networks for two domain sets from Office31 dataset and two ZSL datasets. For comparison, we also included the performance of the minimization solution (*i.e.*, Min).

Change in dimensionality. We test three input/output configurations of the confusion network. Namely, **1.** Equal-dim-confusion(EDC, *see* Fig. 4(a)), where the input and output dimensions are equal, **2.** Low-dim-confusion (LDC, *see* Fig. 4(b)), where the output dimension is lower than the input , and **3.** High-dim-confusion (HDC, *see* Fig. 4(c)) where the output dimension is higher than the input. For the two UDA sets, the output dimensions of the LDC, EDC and HDC models are 128, 256 and 512, respectively. For the two ZSL sets they are 384, 512 and 768, respectively. We observe that the EDC and HDC structures perform better than the LDC structure. This is intuitive as going down with the dimension might lead to loss of information. Overall, we observe that the EDC structure is marginally better than the HDC network.

Use of Batch Normalization. For the network Confusion-with-Batch-Norm (BNC, *see* Fig. 4(d)), we use an additional Batch Normalization layer after the EDC model. In our preliminary experiments without statistical accumulation, we observed that confusion networks performing marginally better with Batch-Normalization. By design, the Batch Normalization layers attempt to keep a stable output feature distribution. This stability can be a favorable condition for statistical alignment. However, when statistical accumulation is used we observe that the proposed EDC model outperforms the BNC network (*see* Table 4). We consider this to be an indication that the proposed statistical accumulation is a better regularizer for the proposed statistical alignment.

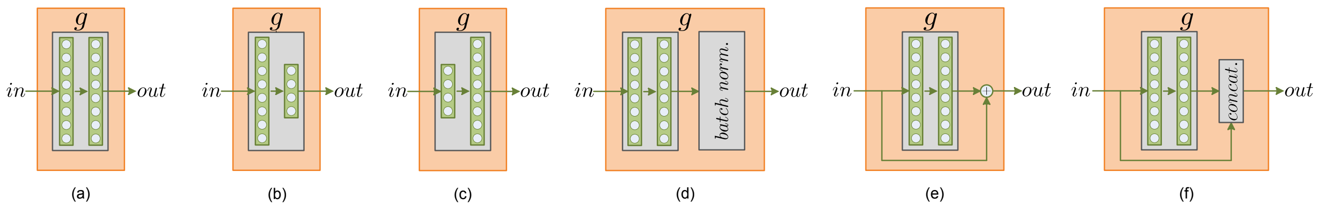


Figure 4. Schematic diagrams of the confusion networks.

Experiment	Min	EDC	LDC	HDC	BNC	Re.C	De.C
A2D	65.5	69.3	64.1	67.3	66.5	68.3	69.7
W2A	48.4	53.9	49.5	52.0	43.3	52.7	50.0
Awa1	53.1	51.9	47.8	52.4	23.4	54.5	40.8
Cub	48.8	50.0	44.8	50.3	22.5	50.6	17.4

Table 4. The performance for various confusion networks structures (Equal-dim-confusion (EDC), Low-dim-confusion (LDC), High-dim-confusion (HDC), Confusion-with-Batch-Norm. (BNC), Residual-confusion (Re.C), and Dense-confusion (De.C)) given in Fig. 4. For comparison, we also include the performance of the KL-minimization.

Solution	Awa1	Awa2	Cubs	Sun
SAE [6]	53.0	54.1	33.3	40.3
ZKL [14]	70.1	70.5	51.7	61.7
Cls. Prot. [5]	69.9	-	54.3	63.3
CLSW [12]	68.2	-	57.3	60.8
Min	61.7	62.3	52.8	54.2
Min-Max	64.1	69.5	56.6	58.8

Table 5. Comparison of the proposed ZSL solution (Min-Max) on conventional ZSL for the proposed splits in [13]. Although our solution does not outperform state-of-the-art solutions in this conventional protocol, we observe that the proposed min-max consistently outperforms the minimization solution.

Use of Skip Connections. Here we explore two skip connection based confusion network structures. Namely, **1.** Residual-confusion (Re.C) (*see* Fig. 4(e)) and **2.** Dense-confusion (De.C) (*see* Fig. 4(f)). Similar to ResNet [3], we perform an addition of the skip connection for the Re.C model. We use a concatenation of the skip connection for De.C in the spirit of the DenseNet [4] and the Inception module [9]. In comparison to our EDC and De.C networks, we observe a stable performance improvement from the Re.C model. Therefore, for all our reported results we use Re.C confusion network.

6. Convergence of the min-max and stopping criteria

We report the performance of min and min-max solutions after a fixed number of training iterations. In Fig. 5, we show the min-max alignment training loss for a UDA and a ZSL set. Similar convergence curves are observed in other experiments.

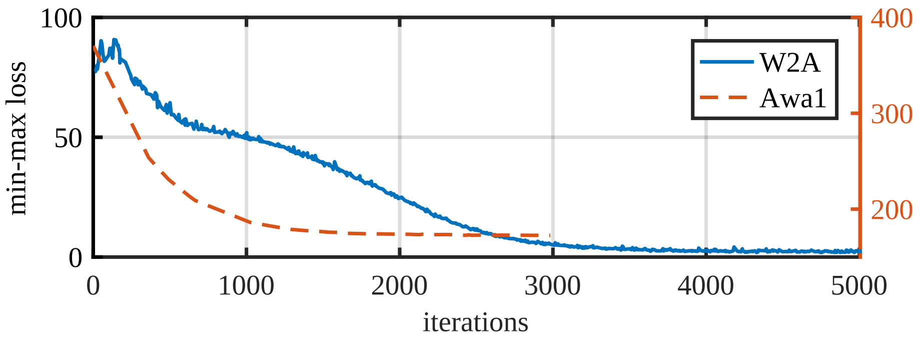


Figure 5. Convergence of min-max training for the W2A and Awa1 experiment sets.

7. Conventional ZSL Performance

It is our understanding that the generalized protocol (*i.e.*, GZSL [13, 11]) is more recent and represents a more challenging problem than the conventional ZSL. That said, we evaluated the min-max and min solution on conventional ZSL(*see* Table 5). We achieve competitive result on the CUBs and Awa2 datasets (56.9% and 69.5%, respectively). As a pointer, the min solution achieves 52.8% and 62.3% for these datasets, respectively. On the Awa1, the min-max solution, outperforms the min solution by a large gap but comes short in comparison to the SOTA ZSL solutions (our result reads as 64.1%).

References

[1] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 248–255, 2009. 2

[2] Yaroslav Ganin and Victor Lempitsky. Unsupervised domain adaptation by backpropagation. In *International Conference on Machine Learning*, pages 1180–1189, 2015. 2

[3] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016. 5

[4] Gao Huang, Zhuang Liu, Laurens van der Maaten, and Kilian Q Weinberger. Densely connected convolutional networks. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2261–2269. IEEE, 2017. 5

[5] Huajie Jiang, Ruiping Wang, Shiguang Shan, and Xilin Chen. Learning class prototypes via structure alignment for zero-shot recognition. In *Proc. European Conference on Computer Vision (ECCV)*, 2018. 5

[6] Elyor Kodirov, Tao Xiang, and Shaogang Gong. Semantic autoencoder for zero-shot learning. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3174–3183, 2017. 5

[7] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Proc. Advances in Neural Information Processing Systems (NIPS)*, pages 1097–1105, 2012. 2

[8] Rui Shu, Hung Bui, Hirokazu Narui, and Stefano Ermon. A DIRT-t approach to unsupervised domain adaptation. In *International Conference on Learning Representations*, 2018. 2

[9] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1–9, 2015. 5

[10] Dmitry Ulyanov, Andrea Vedaldi, and Victor Lempitsky. Instance normalization: The missing ingredient for fast stylization. *arXiv preprint arXiv:1607.08022*, 2016. 2

[11] Yongqin Xian, Christoph H Lampert, Bernt Schiele, and Zeynep Akata. Zero-shot learning-a comprehensive evaluation of the good, the bad and the ugly. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 2018. 5

[12] Yongqin Xian, Tobias Lorenz, Bernt Schiele, and Zeynep Akata. Feature generating networks for zero-shot learning. *Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018. 5

[13] Yongqin Xian, Bernt Schiele, and Zeynep Akata. Zero-shot learning-the good, the bad and the ugly. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3077–3086, 2017. 5

[14] Hongguang Zhang and Piotr Koniusz. Zero-shot kernel learning. *Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018. 5