Domain Generalization with Correlated Style Uncertainty

The supplementary materials provide the pseudo-code for implementing the Correlated Style Uncertainty (CSU) model. To guarantee the reliability of reporting results, we also conduct training stability analysis on PACS dataset. We conduct an ablation experiment to analyze the position selection effects on the Duke-Market1501 instance retrieval task. Furthermore, we visualize the extracted feature using t-SNE projection, which proves that CSU can help with extracting domain-invariant feature representations.

1. Pseudo Code

Here, we provide the pseudo-code for our CSU model. As we can observe, this code is relatively easy to implement and can be encoded into most current models. Note that we do not use the backpropagation of the normal PyTorch Eigh function for eigenvalue decomposition to avoid instability during training. This is because the gradient calculation relies on the smallest value of the eigenvalue difference $\frac{1}{min(\lambda_i - \lambda_j)}$ [1].

```
# Given eps, alpha, p=0.5
# Input: x:B*C*H*W
# Output: x:B*C*H*W
def decompose(matrix):
    with torch.no_grad():
        value, vect = eigh(matrix)
    lmda = sqrt(vect.T@matrix@vect))
    return vect@lmda@vect.T

def forward(x)
    if random < p:
        return x
    mu = mean(x, dim=(2, 3))
    sig = std(x, dim=(2, 3)) + eps
    x_norm = (x - mu) / sig
    corr_mu = decompose(mu.T@mu)
    corr_sig = decompose(sig.T@sig)
    rand_mu = randn_like(mu)@corr_mu
    rand_sig = randn_like(sig)@corr_sig
    inten = Beta(alpha, alpha).sample(N, 1)
    mu = mu + inten*rand_mu
    sig = sig + inten*rand_sig
    x = mu + x_norm*sig
    return x
```

Listing 1. An Pytorch-like pseudo code for CSU

This can induce extremely unstable training, considering that we have many zero eigenvalues, as described in the previous section. We assume that the direction is relatively stable during the training to address this issue. The key for backpropagation is calculating the eigenvalue or variance intensity for the corresponding direction. Thus, we adopt an algorithm that does not pass the gradient through the eigenvector. We show the pseudo implementation in 1

2. Training Reliability

We conduct the training process using the exact configuration on the PACS dataset multi-time. Here we set $\alpha = 0.3$. We perform 20 times of experiments and calculate the performance distribution to test the training stability. The standard deviations of Art, Cartoon, Photo, and Sketch are 0.35, 0.17, 0.12, and 0.30, respectively, and the standard deviation of Average is 0.13. Figure 1 shows the result. We can observe that the standard deviation is relatively low, and the One-Sigma range is (84.90, 85.17), which indicates that the training process is consistent and reliable.

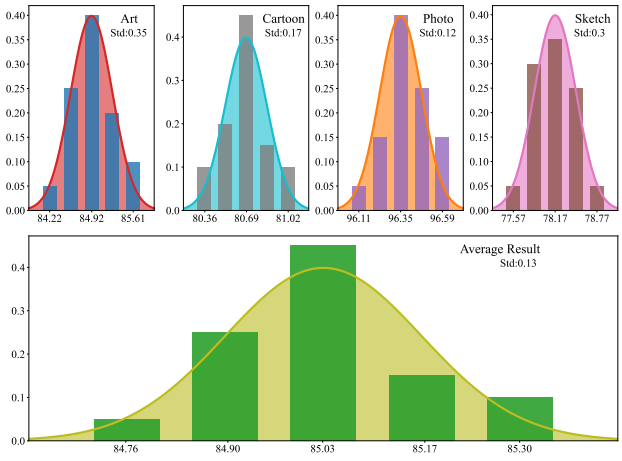

Figure 1. One visualization of training stability. We perform 20 times of experiments and calculate the standard deviations of each category and the average performance. We can observe that the training is stable given the low standard deviation value.

3. Position Selection For Instance Retrieval

We conduct an ablation experiment to analyze the position selection effects on the Duke-Market1501 instance retrieval task. Here we show the influence of different insert-

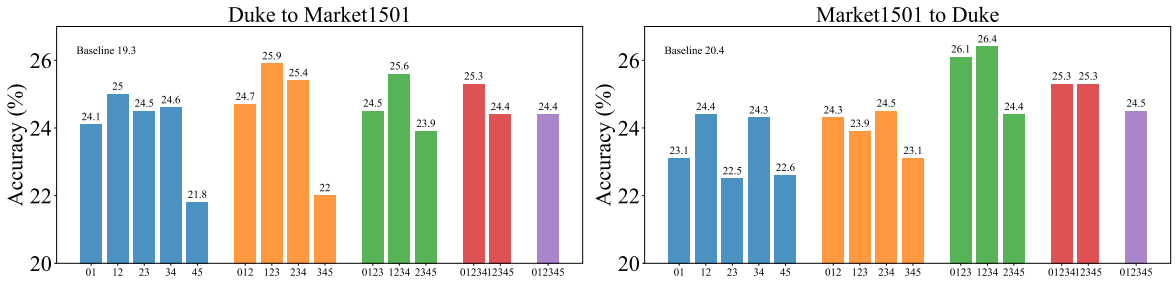

Figure 2. One visualization of different inserting positions on the instance retrieval experiments. We can achieve better performance than reporting by changing the inserting position. This shows the best position configuration might vary by task rather than one fixed conclusion.

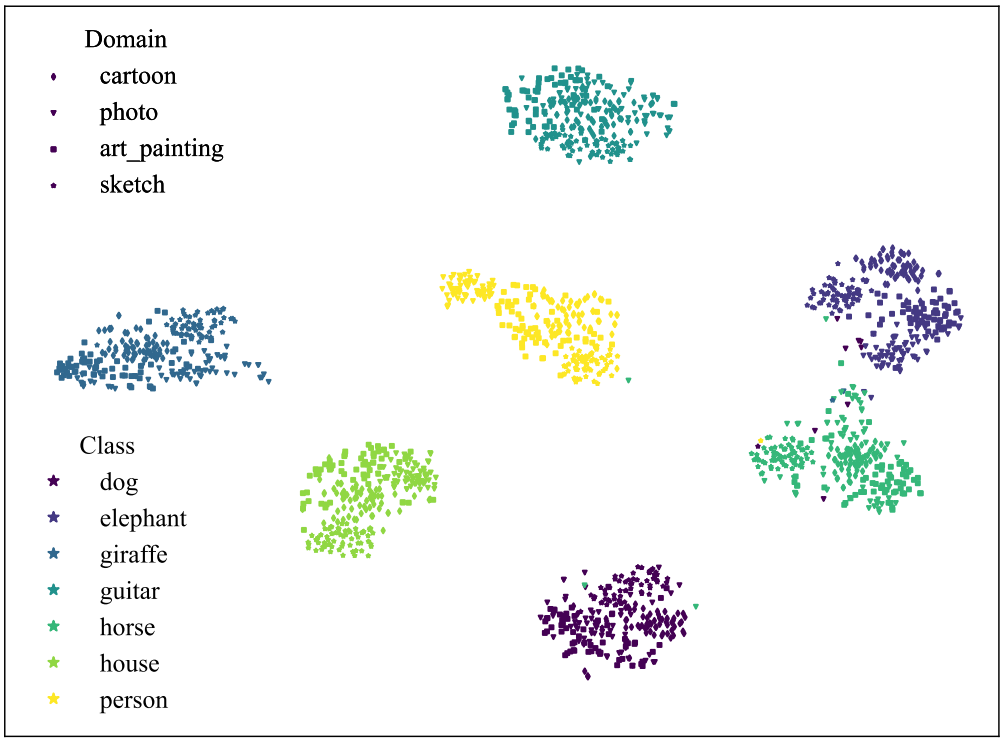

Figure 3. 2-D Visualization of Flattened Feature Maps. We can clearly observe that the trained model can effectively obtain more domain-invariant feature representations. For 4 domains of the PACS dataset, including Art, Cartoon, Photo, and Sketch, we select 64 cases from every category (7 categories in total) under each domain.

ing positions in Figure 2. We can find that overall trends are similar to the classification tasks. Notably, we can achieve impressive improvement compared to the reported result (the "012345" group) by changing position. This indicates that the best position configuration may vary by task rather than one fixed conclusion.

4. Visualization of Flattened Feature Maps

To intuitively understand the effectiveness of our method, we provide the t-SNE visualization map of feature vectors extracted from the trained model. As shown in Figure 3, we can find that with the CSU, the distance between different domains within the same category is small, while the distance between different classes, regardless of the domain, is immense. Therefore, we can show that the trained model can obtain more domain-invariant feature representations, indicating a more vital generalization ability.

References

[1] Catalin Ionescu, Orestis Vantzos, and Cristian Sminchisescu. Matrix backpropagation for deep networks with structured layers. In *Proceedings of the IEEE international conference on computer vision*, pages 2965–2973, 2015. 1