Domain Adaptation for Rare Classes Augmented with Synthetic Samples

Delft University of Technology, Bachelor Seminar of Computer Science and Engineering

Tuhin Das[1]

Robert-Jan Bruintjes[1]

Attila Lengyel[1]

Jan van Gemert[1]

Sara Beery[2]

[1] Computer Vision Lab
Delft University of Technology
Delft, The Netherlands

[2] Computational Vision Lab
California Institute of Technology
California, USA

Abstract

To alleviate lower classification performance on rare classes in imbalanced datasets, a possible solution is to augment the underrepresented classes with synthetic samples. Domain adaptation can be incorporated in a classifier to decrease the domain discrepancy between real and synthetic samples. While domain adaptation is generally applied on completely synthetic source domains and real target domains, we explore how domain adaptation can be applied when only a single rare class is augmented with simulated samples. As a testbed, we use a camera trap animal dataset with a rare *deer* class, which is augmented with synthetic deer samples. We adapt existing domain adaptation methods to two new methods for the single rare class setting: *DeerDANN*, based on the Domain-Adversarial Neural Network (DANN), and *DeerCORAL*, based on deep correlation alignment (Deep CORAL) architectures. Experiments show that DeerDANN has the highest improvement in deer classification accuracy of 24.0% versus 22.4% improvement of DeerCORAL when compared to the baseline. Further, both methods require fewer than 10k synthetic samples, as used by the baseline, to achieve these higher accuracies. DeerCORAL requires the least number of synthetic samples (2k deer), followed by DeerDANN (8k deer).

1 Introduction

Computer vision models generally perform well on datasets where each class is well-represented [14]. However, real-world datasets often have long-tailed distributions [38, 46, 52], which can lead to class imbalances. As a result, the performance of computer vision algorithms on underrepresented classes can be inferior to the performance on well-represented classes [3, 46]. In some domains, recognition of rare classes is especially important, e.g. for the monitoring of rare animals in camera trap datasets, which motivates the investigation of methods for improving rare-class performance. A possible method to reduce the performance discrepancy between imbalanced classes is to oversample the rare classes with new samples [11]. As finding new samples of rare classes can be difficult, synthetic samples can be

used instead. Generating synthetic samples of rare classes is especially attractive as computer vision algorithms have been shown to perform well on real test sets when trained on fully synthetic datasets [12, 17, 31, 47] or on synthetically augmented datasets [16, 36, 37].

As a real-world testbed, in this work we focus on camera trap data containing images of animals captured by heat or motion-activated passive monitoring cameras. These cameras are widely used to monitor biodiversity and animal behavior, and to measure the efficacy of conservation actions. Camera traps collect enormous datasets, up to millions of images from a single network of cameras in a single season. It is inhibitively time-consuming for ecologists to manually label the species seen in the data, thus automated classification of animal species is needed to match the speed of data collection.

The distribution of species in camera trap datasets is long-tailed [3, 4, 5, 6, 8, 33], mimicking the distribution of species in the natural world. In [7], the Caltech Camera Traps (CCT) dataset [2, 3], which contains a rare *deer* class, was augmented with synthetic deer images generated with 3D game engines to reduce the class imbalance and increase the pose and location diversity of the rare class samples. While [7] found that the synthetically augmented training set led to considerably reduced classification errors for the deer class, the performance was still low - average precision for the rare class maxed out at 66% using 100k synthetic samples. It also appeared that there was no overlap between the deep network representations of the real deer and the synthetic deer. As the synthetic deer are not photorealistic and do not have realistic image statistics, there is a domain discrepancy between the real and synthetic deer. Our objective is to minimize the domain discrepancy between real and synthetic deer to improve recognition of the rare deer class.

In this work, we propose using domain adaptation methods to minimize the domain discrepancy between real and synthetic deer. Domain adaptation techniques are used to create a shared feature space between a source and a target domain, such that an algorithm trained on the source domain can be applied directly to the target domain [34]. Domain adaptation has been successfully used to reduce domain discrepancy between real and synthetic samples when training on a synthetic source domain and testing on a real target domain [1, 15, 24, 39, 49]. In our case, the source and target domain are not the training and test set, but the sets of synthetic deer and real deer respectively, both of which are contained in the training set. Additionally, other animal classes are included in the dataset, represented by real samples only, which need to be classified accurately as well.

We explore two methods of applying domain adaptation on synthetic and real deer samples. The first method, which we refer to as DeerDANN, is based on the Domain-Adversarial Neural Network (DANN) [19] and incorporates a domain discriminator in a classifier with the task to guess which domain a sample belongs to. While DANNs originally apply the discriminator to samples from each class, we modify the network such that the discriminator only receives deer samples and guesses whether a sample is a real deer or synthetic deer. This modification removes the unnecessary domain confusion loss of non-deer samples. By maximizing the discriminator loss and minimizing the classification loss, the classifier learns to extract domain-invariant features for the deer class, while the features from all classes retain class discriminability.

The second method, which we refer to as DeerCORAL, incorporates a correlation alignment (CORAL) loss [41] in a classifier, as in Deep CORAL architectures [40]. The CORAL loss represents the distance between the second-order statistics of the source and target domain. By minimizing the CORAL loss and classification loss, the domain discrepancy is reduced while the features retain class discriminability. As synthetic samples are only available for the rare class, we organise the real and synthetic data as source and target domains in a

different manner than for the original DANN and Deep CORAL methods. Fig. 3 presents the schematic architectures of our methods as well as the new domain organisation.

The main contribution of this work is that we adapt the DANN and Deep CORAL domain adaptation techniques to our setting, where synthetic target domain data is only available for a single rare class. As a result, we significantly improve the classification accuracy of the rare deer class in the CCT dataset [2, 3] when compared to a baseline model from [7], without notably reducing performance on the other classes in the dataset. Additionally, we show that the number of synthetic samples needed can be greatly reduced when applying domain adaptation on the synthetic samples.

2 Related Work

Domain Adaptation Domain adaptation is used to make models learn transferable representations from a source dataset, to enable making inference on a target dataset directly [48]. Some methods align the distributions of the source and target domains by minimizing some distance between the source and target distributions, such as the Maximum Mean Discrepancy (MMD) [30, 43, 50] or the correlation alignment (CORAL) [35, 40, 41]. Domain-Adversarial Neural Networks (DANNs) [18, 19] and Adversarial Discriminative Domain Adaptation (ADDA) [44] confuse a discriminator that classifies sample features to the source or target domain to generate domain-invariant features. Other methods apply Generative Adversarial Networks (GANs) to generate samples that are similar to target samples while retaining annotations from source samples [10, 27, 29]. A final class of methods applies encoder-decoders to learn domain-invariant feature representations and to perform data reconstruction [21, 53].

Generally, domain adaptation methods are applied between a source training set and a target test set, where examples of all classes of interest are present in both sets. Our work differs in that respect, as we attempt to apply domain adaptation on real and simulated samples within a single class, without affecting performance on the other classes.

Domain Adaptation for Synthetic Data Domain adaptation has often been applied for transfer learning between a synthetic source domain and a real target domain. An autoencoder-based approach for lane detection has been used in [20], where an autoencoder was trained in unsupervised and semi-supervised settings on synthetically generated road images. Deep Generative Correlation Alignment Networks (DGCANs) [35] can generate new synthetic samples by combining a 3D CAD synthetic domain and a real domain for improving object detection. Pixel-level domain adaptation on simulated objects using a discriminator and a generator has been used to train a robotic arm to grasp real-world objects [9]. A GAN-based image-to-image translation method called CycleGAN [51] has been used to transform synthetic crowd images into realistic images to improve crowd counting [49]. Style transfer and adversarial learning have been combined to learn monocular depth estimation from synthetic images [1]. Finally, the approach of [24] improves face recognition with only single samples available per person by generating synthetic poses of existing samples and performing adversarial domain adaptation on the synthetic data.

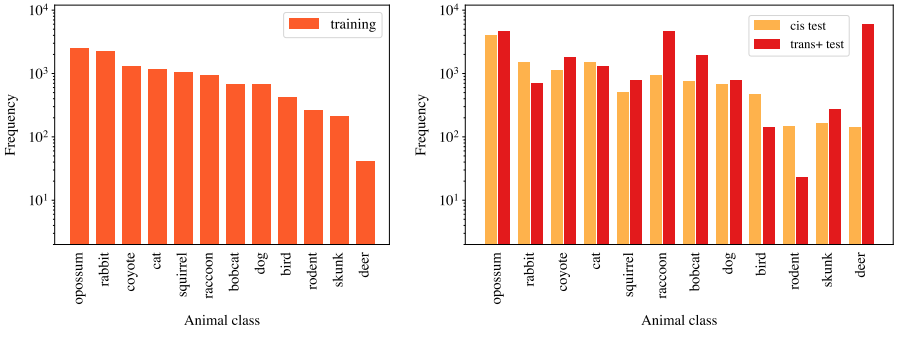
Many of these methods [1, 9, 20, 24] utilize a discriminator, originating from DANNs, to introduce domain confusion between real and synthetic samples. While DANNs fall into the supervised domain adaptation category, DGCANs [35] minimize CORAL loss between real and synthetic samples, which is an unsupervised domain adaptation technique. To test both supervised and unsupervised methods in our setting, we investigate how DANNs and CORAL loss can be applied to synthetic samples of a single rare class.

3 Dataset

We use the Caltech Camera Traps (CCT) dataset [3] as training and test data. The original CCT dataset contains 243,187 images of 30 different animal classes captured by heat- or motion-triggered cameras across 140 wildlife locations. We split the dataset following the CCT-20 data split as described in [3], containing 57,868 images with accompanying bounding boxes of 15 animal classes across 20 camera locations.

The CCT-20 split was originally proposed to evaluate the generalization of camera trap classification and detection models on test sets with different image statistics. To achieve different training and test distributions, the CCT-20 data set is split across camera locations using two partitions: **cis** images are from camera locations seen in the training set and **trans** images are from locations unseen in the training set to test generalization to new locations.

The CCT-20 data set is split into five subsets: training, cis validation, cis test, trans validation, and trans test. The CCT-20 training set clearly shows the long-tailed distribution of animal classes, with the deer class as the rarest class in the training data with only 41 samples (see Fig. 1(a)).



(a) CCT-20 training

(b) CCT-20 cis and trans+ test

Figure 1: The animal class frequencies in the CCT-20 training set (a) and test sets (b). The training set shows the long-tailed nature of the animal class distribution, in which the deer class is the rarest class with only 41 samples.

As the trans sets do not contain any deer samples in the CCT-20 split, we augment the trans sets with deer samples from other CCT locations to become **trans+** sets (see Fig. 1(b)), as in [7]. Following [7], we also remove the *badger*, *fox*, *empty* and *car* classes from the data sets to retain only images containing animals (focusing on classification as opposed to detection), and to isolate the deer class as the single rare class to prototype our approach.

As synthetic data we use the simulated deer images that were generated for the CCT-20 dataset in [7]. The synthetic deer are created using the Unity 3D game development engine and have accompanying bounding boxes. To improve the classification performance of the deer class, the simulations are varied in models, pose, lighting, and day or night-time rendering. Fig. 2 shows examples of real and synthetic deer images.

4 Methods

We adapt two domain adaptation techniques, DANN [19] and Deep CORAL [40], to our setting where synthetic data is available for only a single rare class. As our source and target domains are subsets of the training set, with no target data in the test set, we set up the domains differently than with the standard DANN and Deep CORAL methods. Here we describe our proposed adaptations and how we organise the data as source and target domains.

(a) Real day (b) Real night (c) Simulated day (d) Simulated night

Figure 2: Real deer images from the CCT-20 training set (a, b) and synthetic deer images generated with the Unity 3D game engine (c, d) used to augment the rare deer class.

4.1 DeerDANN

The first method we consider uses a deep network with a feature extractor F, domain discriminator D, and label classifier C, similar to the DANN architecture [19]. The discriminator is given the task to classify all deer features from F as a source (synthetic) or target (real) deer sample. The source domain S contains the full CCT-20 training set and is extended with synthetic deer. The target domain T contains the 41 real deer oversampled 50 times to end up with 2050 real deer.

During training, samples from both domains go through F to generate a feature vector f. All deer features f are sent to D and D guesses which domain each deer f belongs to, leading to a domain confusion loss $\mathcal{L}_D$ for source and target deer samples. C classifies all f coming from S, leading to a classification loss $\mathcal{L}_C$. The network is trained end-to-end using the composite loss

$$\mathcal{L}_{\text{sum}} = \sum_{i \in S} \mathcal{L}_C^i + \sum_{i \in \delta(S \cup T)} \mathcal{L}_D^i, \tag{1}$$

where $\delta(\cdot)$ represents all deer features from a domain. The parameters of F and C are optimized to minimize $\mathcal{L}_C$. By inserting a Gradient Reversal Layer (GRL) [18] between F and D, F is updated to maximize $\mathcal{L}_D$ while D is updated to minimize $\mathcal{L}_D$. As a result the network adversarially learns to generate deer features that are domain-invariant and are discriminative from other classes. Fig. 3 shows the proposed architecture of DeerDANN. Note that Fig. 3 includes the training set in the target domain for simplicity, which does not make a difference for training DeerDANN.

4.2 DeerCORAL

The second method we consider, DeerCORAL, utilizes correlation alignment like Deep CORAL architectures [40] by incorporating a CORAL loss that represents the distance between the second-order statistics of the source and target domain [41]. The CORAL loss is computed over the classification logits of a deep network with

$$\mathcal{L}_{\text{CORAL}} = \frac{1}{4d^2} \|C_S - C_T\|_F^2, \tag{2}$$

where $\|\cdot\|_F^2$ denotes the squared Frobenius norm, C_S and C_T are the covariance matrices of the source and target data respectively, and d is the dimension of the deep layer activation.

S and T are set up similarly as for DeerDANN, except that T also contains the training set. T is constructed in this manner to resemble S in class priors, but with the features of real deer instead of synthetic deer. During training, source samples are used to compute a classification loss $\mathcal{L}_C$. Further, a CORAL loss is computed using the source and target batch with Eq. (2).

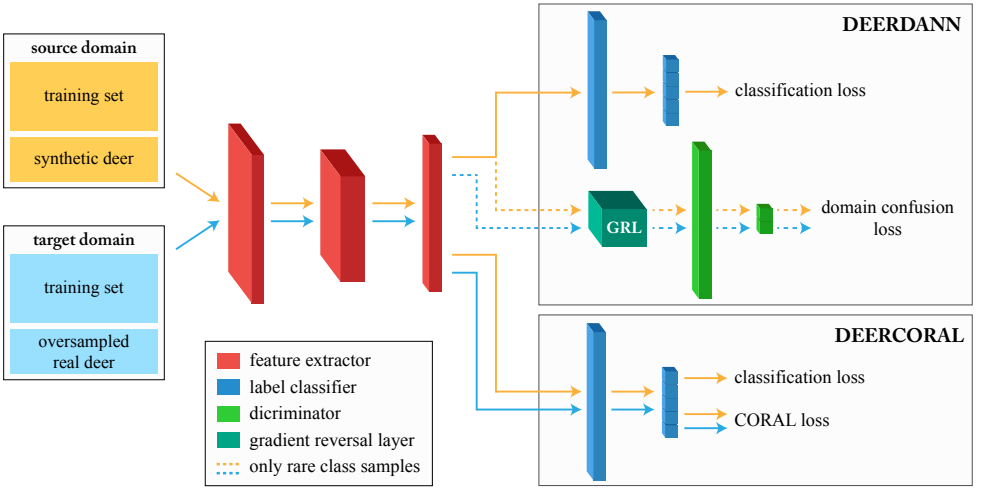


Figure 3: Proposed architectures for adapting either DANN [19] or Deep CORAL [40] to synthetic samples of a single rare class. **(a) DeerDANN**. A label classifier is used to predict the category of source examples only, while the discriminator is used to predict the domain of deer samples. Classification loss is minimized, while domain confusion loss is maximized by the Gradient Reversal Layer (GRL) to generate domain-invariant deer features. Note that the target domain does not need to contain the training set for DeerDANN. **(b) DeerCORAL**. A feature extractor and label classifier are enhanced with a CORAL loss term, computed with second-order statistics of the source and target domain. CORAL loss and classification loss of source features are both minimized.

The network is trained end-to-end using the composite loss

$$\mathcal{L}_{\text{sum}} = \sum_{i \in S} \mathcal{L}_C^i + \lambda \cdot \mathcal{L}_{\text{CORAL}}, \tag{3}$$

where λ is a hyperparameter controlling the trade-off between classification loss and CORAL loss. By minimizing the composite loss $\mathcal{L}_{\text{sum}}$, the network learns to extract features with similar statistics from both the source and target domains, but with discriminative distributions for different classes. See Fig. 3 for the overall DeerCORAL architecture.

5 Experiments

In our experiments we perform cropped bounding box classification of animal species in the CCT-20 data set. Next to the CCT-20 images, we use a pool of 5k day and 5k night simulated deer images from where we sample subsets of synthetic deer for experiments. All CCT-20 images and synthetic images are cropped to the provided bounding boxes and are rescaled to 299×299 pixels.

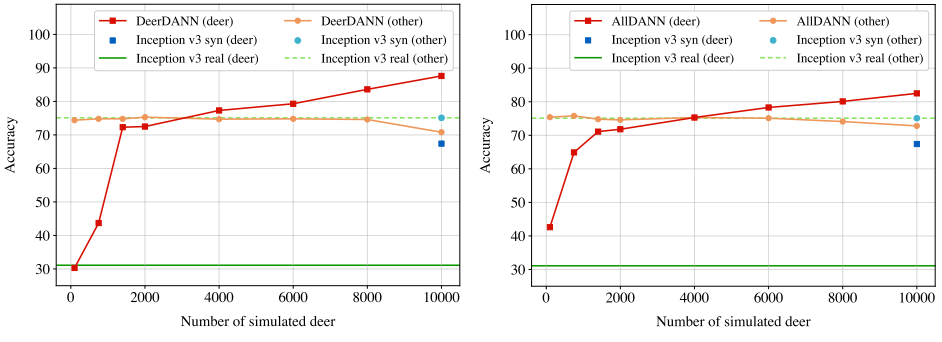
As baseline we use an Inception v3 model [42] pre-trained on ImageNet [14] with an initial learning rate of 0.0045, RMSprop with 0.9 momentum, and horizontal flipping, color jitter, and blur as data augmentation, following [7]. The Inception model is fine-tuned on the CCT-20 training set, as well as on the training set augmented with 10k synthetic deer images. We refer to these models as *Inception v3 real* (real samples only) and *Inception v3 syn* (real and synthetic samples).

5.1 Effect of Number of Simulated Deer on DeerDANN Performance

For DeerDANN we use an ImageNet pre-trained ResNet50 model [23] without the last layer as feature extractor F. Fully connected layers are used as discriminator D (1024 - 1024 - 2) and as label classifier C (1024 - 12). During training we use an initial learning rate of 10^{-5} and the Adam optimizer [28] with L2 regularization. Random cropping, color jitter, and horizontal flipping are applied on all real and synthetic samples as data augmentation. The number of simulated deer is varied from 100 to 10k samples and models are selected using the trans+ validation set.

Additionally, we train a similar model with the same settings, where samples from all classes are sent to the discriminator instead of only deer samples. We call this model AllDANN and use this model to investigate whether isolated domain adaptation on the deer class is favourable to domain adaptation on all classes. For AllDANN the target domain also contains the training set, as shown in Fig. 3.

For 1400 or more synthetic deer, both DeerDANN and AllDANN have higher accuracies than the Inception v3 syn baseline, trained with 10k synthetic deer, on the trans+ deer class (see Fig. 4). The performance of both models for the other classes on average is similar to the baselines, for 8k or fewer simulated deer. When adding up to 10k synthetic deer, the performance of the deer class increases considerably, at the cost of a larger drop in other classes. In that case, the large deer accuracy increase is most likely due to the large deer class prior, which is four times larger than the prior of the second most common *opossum* class.



(a) DeerDANN (b) AllDANN

Figure 4: Learning curves of DeerDANN and AllDANN for various numbers of simulated deer ranging from 100 to 10k. Accuracy is measured on the trans+ test set. For 1400 or more simulated deer, both methods have higher deer accuracies than the Inception v3 syn baseline.

Between DeerDANN and AllDANN, DeerDANN seems to perform better on the trans+ deer class for 4k or more simulated samples. Thus, it seems beneficial to only channel the deer features to the discriminator instead of all features, when performing domain adaptation on a single class.

5.2 Effect of Number of Simulated Deer on DeerCORAL Performance

For DeerCORAL we again use an ImageNet pre-trained ResNet50 model as feature extractor and two fully connected layers as label classifier (1024 - 12). Training is performed with an initial learning rate of 10^{-5}, λ equal to 0.5, Adam with L2 regularization and the same data augmentation techniques as for DeerDANN and AllDANN. The last two layers are trained with a learning rate 10 times larger than the learning rate of the other layers, following [40].

Similar to DeerDANN and AllDANN, DeerCORAL performs better than the baselines on the trans+ test deer class when using 1400 or more simulated deer (see Fig. 5). The average accuracy of other classes seems to drop faster than for the DANN models. This drop can be avoided by using fewer deer samples, as DeerCORAL already shows significant improvement with 2k simulated deer.

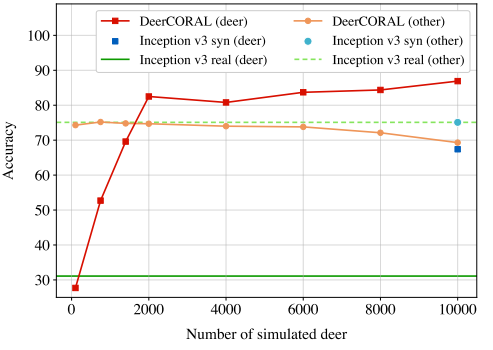


Figure 5: Learning curves of DeerCORAL for various numbers of simulated deer ranging from 100 to 10k. Accuracy is measured using the trans+ test set. For 1400 or more simulated deer, DeerCORAL has higher deer accuracies than the Inception v3 syn baseline.

5.3 Comparison of DeerDANN, AllDANN, and DeerCORAL

From each method we select the model with the highest deer accuracy and a loss of at most 1% in trans+ other class accuracy. These models are DeerDANN with 8k synthetic deer, AllDANN with 6k deer, and DeerCORAL with 2k deer. We compare these models with the Inception v3 baselines, as well as an ImageNet pre-trained ResNet50 model, fine-tuned on the CCT-20 training set with 10k synthetic deer, which we call *ResNet50 syn*. ResNet50 syn is trained exactly like DeerDANN and AllDANN, and serves as an ablation model to study the performance of just the ResNet50 backbone. We compare the model accuracies for the deer class as well as for the other classes on the cis and trans+ test sets (see Table 1).

Table 1: A comparison between AllDANN (6k syn deer), DeerDANN (8k syn deer), DeerCORAL (2k syn deer), the Inception v3 baselines (0 and 10k syn deer), and an ablation ResNet50 model (10k syn deer) on the cis and trans+ test sets.

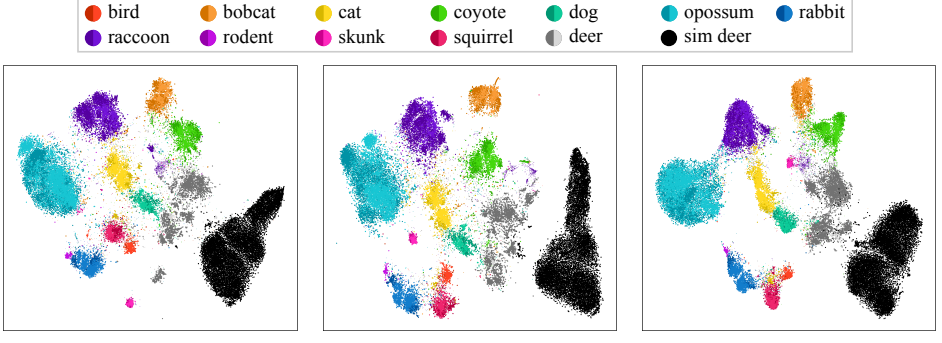
	trans+ deer	cis deer	trans+ other (avg.)	cis other (avg.)
Inception v3 real	31.1	51.7	75.1	89.5
Inception v3 syn	67.4	68.3	75.1	91.0
ResNet50 syn	42.3	58.7	74.1	90.0
AllDANN	78.3	94.2	75.1	89.4
DeerDANN	83.6	96.4	74.6	89.3
DeerCORAL	82.5	97.1	74.7	90.0

First of all, Inception v3 syn performs better than ResNet50 syn in each category. Thus, the improvements of the domain adaptation models are not caused by the ResNet50 feature extractor. Further, all domain adaptation models have higher deer accuracies than the Inception baselines. DeerDANN has the largest accuracy improvement in the trans+ deer class of 24.0% versus 16.2% improvement of AllDANN and 22.4% of DeerCORAL. For the cis deer class DeerCORAL performs best, with 42.2% improvement versus 37.9% and 41.1% of AllDANN and DeerDANN respectively. All models perform quite similarly for the other trans+ classes,

but slightly less good on the other cis classes. All domain adaptation models require fewer than 10k deer to achieve higher deer accuracies than the Inception baselines, which were trained using 10k synthetic deer. In particular, DeerCORAL requires few simulated deer samples (2k) versus 8k of DeerDANN and 6k of AllDANN.

5.4 Network Feature Visualization

In [7], the CCT-20 classifier learned the deer class bimodally, as the network features of real and synthetic deer did not overlap. Our expectation is that features generated by our models are similar for the real and synthetic deer samples. We perform 200-dimensional PCA and t-SNE [45] afterwards on the feature activations of the last pre-logit layer to visualize the features, as seen in Fig. 6.



(a) AllDANN features (b) DeerDANN features (c) DeerCORAL features

Figure 6: Last pre-logit layer feature representations of all domain adaptation models of CCT-20 cis and trans+ test samples and 10k simulated deer. The simulated deer (black) and real deer (gray) are not clustered together, which indicates that the models learn the deer class bimodally. Small and large dots respectively show incorrect and correct classifications. Different hues of the same color represent trans+ samples (dark) and cis samples (light).

Surprisingly, DeerDANN and AllDANN appear to cluster the real and synthetic deer separately. Even when a discriminator is applied, the classifiers still learn the real and synthetic deer bimodally. DeerCORAL also learns the real and simulated deer bimodally, which is not surprising as DeerCORAL only minimizes the second-order statistics between the source and target features. Even though the real and synthetic features do not overlap, applying domain adaptation still improves the classification of the rare deer class.

6 Discussion

We present two different methods of applying domain adaptation to improve the classification accuracy of a rare class using simulated samples of that class. The first method (DeerDANN), based on the Domain-Adversarial Neural Network (DANN), incorporates a discriminator in a classifier to generate domain-invariant features. We modify the DANN method such that only samples of the rare class are sent to the discriminator, to perform domain adaptation on the rare class only. The second method (DeerCORAL) incorporates a correlation alignment loss in a classifier, to align the second-order statistics of the source and target domain.

From our experiments we conclude the following points. First of all, both DeerDANN and DeerCORAL have higher deer classification accuracies and similar average accuracies for other classes, when compared to our baseline network. Second, DeerDANN has the largest

improvement in deer classification in unseen locations, which justifies sending only deer samples to the discriminator. Third, both models require fewer simulated samples to achieve considerably better deer accuracies than the baseline. DeerCORAL especially only requires 2k samples versus 10k samples used by the baseline.

Regardless of the improved deer classification using domain adaptation, the network features of the real and synthetic deer surprisingly still do not overlap when visualizing the features in two dimensions. The question remains whether features of synthetic and real samples of a single rare class can be made to overlap using different domain adaptation techniques or other methods.

7 Responsible Research

This research is conducted while keeping in mind the ethical implications and reproducibility of this work. Here we review to what extent our work and the data are ethically just. Further, we discuss how we attempt to make our results reproducible for any interested parties.

7.1 Ethical Implications

Data collection and processing can often be sources of ethical questions. The images in the original CCT dataset were captured by automatic cameras and were labeled by experts [3]. Nevertheless, we can be critical of the bounding box annotations, as those annotations were collected by Amazon Mechanical Turk (MTurk) workers. MTurk is an online marketplace where digital tasks can be performed by workers around the world for a fee [26]. However, in 2018 MTurk workers only had a median hourly salary of $2/h, and only 4% of MTurk workers earned more than $7.25/h [22]. In that sense, the use of MTurk is quite unethical. However, as long as job posters offer proper hourly wages for the jobs, MTurk can be used without any ethical violations. Especially since MTurk workers perform the jobs voluntarily.

As this work is part of the field of machine learning (ML), we also consider how ethical it is to create models that can perform tasks faster than humans can perform. The fact that classifying large datasets is time-consuming for humans, is widely used as an argument to build faster and more efficient ML models. One could argue that ML is effectively being used to minimize man-hours, which in turn could lead to fewer work opportunities for humans. In our particular case, our model indeed reduces labor-intensive labeling work. But labeling of animals in camera trap datasets is often done by experts as in [3]. Thus, experts could save time by employing such ML models and could focus more on analyzing the generated data.

Finally, we consider whether any conflicts of interest play a role in this research. It is important to note that the computational resources for this work are provided by the Microsoft AI for Earth grant. The interests of Microsoft are transparently defined, given that the AI for Earth grants are provided to "support projects that use AI to change the way people and organizations monitor, model, and manage Earth's natural systems." [32]. Our work coherently fits within this viewpoint of Microsoft, as we attempt to use AI for analyzing animal diversity more efficiently using camera trap datasets. As our interests align with the interests of Microsoft, this research has not suffered from any conflicts of interest.

7.2 Experiment Reproducibility

A vital aspect of ML research is the reproducibility of experiments. A recurring problem is when results of a publication can not be reproduced due to missing source code or unknown model training conditions [25]. To properly compare a method with an existing benchmark, it should be possible to exactly replicate the benchmark performance. In our case, we managed

to reproduce the results from [7] as we had access to all necessary model information and all data. Thus, we can ensure that we justly compare our models to a proper benchmark, which leads to trustworthy results.

As deep learning research often is engaged with training deep networks, we consider to what extent deep network results are reproducible and how the reproducibility can be improved. Deep networks are often initialized with random values and are optimized according to deterministic equations. It is essential to store the random seed that is used for training, as the seed determines the initialization of the network weights. Further, the architecture and all used parameters should be stored for each experiment. With all this information published, it should be possible to recreate comparable results with a self-written model. But to recreate the exact results, as desired, the source code should be published along with the parameters and the random seed.

We take various steps to ensure the reproducibility of our work. For our experiments, a summary of all training settings is stored per run. We use a random seed of 0 for all experiments, to ensure that randomness does not play a role in finding the best initial weights. Further, we plan to publish the source code so that anyone can reproduce our exact results.

However, there is one problem that can counteract the reproducibility of our work and other ML work. The use of GPUs introduces some randomness in training, which can not be controlled by a random seed [13]. Thus, it is still possible that models trained with the exact same code, parameters, and seed give slightly different results. This variation in results can be mitigated by using CPUs or by taking average results of multiple runs.

Acknowledgements

Computational resources were provided by Microsoft AI for Earth.

References

[1] Amir Atapour-Abarghouei and Toby P. Breckon. Real-Time Monocular Depth Estimation Using Synthetic Data With Domain Adaptation via Image Style Transfer. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2800–2810, 2018.

[2] Sara Beery. Caltech Camera Traps (CCT). https://beerys.github.io/CaltechCameraTraps/, 2018.

[3] Sara Beery, Grant Van Horn, and Pietro Perona. Recognition in Terra Incognita. In *Proceedings of European Conference on Computer Vision (ECCV)*, Lecture Notes in Computer Science, pages 472–489, 2018.

[4] Sara Beery, Dan Morris, and Pietro Perona. The iWildCam 2019 Challenge Dataset. *arXiv:1907.07617*, 2019.

[5] Sara Beery, Grant van Horn, Oisin Mac Aodha, and Pietro Perona. The iWildCam 2018 Challenge Dataset. *arXiv:1904.05986*, 2019.

[6] Sara Beery, Elijah Cole, and Arvi Gjoka. The iWildCam 2020 Competition Dataset. *arXiv:2004.10340*, 2020.

[7] Sara Beery, Yang Liu, Dan Morris, Jim Piavis, Ashish Kapoor, Markus Meister, Neel Joshi, and Pietro Perona. Synthetic Examples Improve Generalization for Rare Classes.

In *Proceedings of IEEE Winter Conference on Applications of Computer Vision (WACV)*, pages 852–862, 2020.

[8] Sara Beery, Arushi Agarwal, Elijah Cole, and Vighnesh Birodkar. The iWildCam 2021 Competition Dataset. *arXiv:2105.03494*, 2021.

[9] K. Bousmalis, A. Irpan, P. Wohlhart, Y. Bai, M. Kelcey, M. Kalakrishnan, L. Downs, J. Ibarz, P. Pastor, K. Konolige, S. Levine, and V. Vanhoucke. Using Simulation and Domain Adaptation to Improve Efficiency of Deep Robotic Grasping. In *Proceedings of IEEE International Conference on Robotics and Automation (ICRA)*, pages 4243–4250, 2018.

[10] Konstantinos Bousmalis, Nathan Silberman, David Dohan, Dumitru Erhan, and Dilip Krishnan. Unsupervised Pixel-Level Domain Adaptation With Generative Adversarial Networks. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3722–3731, 2017.

[11] Mateusz Buda, Atsuto Maki, and Maciej A. Mazurowski. A systematic study of the class imbalance problem in convolutional neural networks. *Neural Networks*, 106:249–259, 2018.

[12] Wenzheng Chen, Huan Wang, Yangyan Li, Hao Su, Zhenhua Wang, Changhe Tu, Dani Lischinski, Daniel Cohen-Or, and Baoquan Chen. Synthesizing Training Images for Boosting Human 3D Pose Estimation. In *Proceedings of Fourth International Conference on 3D Vision (3DV)*, pages 479–488, 2016.

[13] Torch Contributors. Reproducibility — PyTorch documentation. https://pytorch.org/docs/stable/notes/randomness.html, 2019. Accessed: 2021-06-21.

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

[15] Aysegul Dundar, Ming-Yu Liu, Ting-Chun Wang, John Zedlewski, and Jan Kautz. Domain Stylization: A Strong, Simple Baseline for Synthetic to Real Image Domain Adaptation. *arXiv:1807.09384*, 2018.

[16] Maayan Frid-Adar, Idit Diamant, Eyal Klang, Michal Amitai, Jacob Goldberger, and Hayit Greenspan. GAN-based synthetic medical image augmentation for increased CNN performance in liver lesion classification. *Neurocomputing*, 321:321–331, 2018.

[17] Adrien Gaidon, Qiao Wang, Yohann Cabon, and Eleonora Vig. VirtualWorlds as Proxy for Multi-object Tracking Analysis. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4340–4349, 2016.

[18] Yaroslav Ganin and Victor Lempitsky. Unsupervised Domain Adaptation by Backpropagation. In *Proceedings of International Conference on Machine Learning (ICML)*, pages 1180–1189, 2015.

[19] Yaroslav Ganin, Evgeniya Ustinova, Hana Ajakan, Pascal Germain, Hugo Larochelle, François Laviolette, Mario Marchand, and Victor Lempitsky. Domain-Adversarial Training of Neural Networks. In *Domain Adaptation in Computer Vision Applications*, Advances in Computer Vision and Pattern Recognition, pages 189–209. 2017.

[20] Noa Garnett, Roy Uziel, Netalee Efrat, and Dan Levi. Synthetic-to-Real Domain Adaptation for Lane Detection. *arXiv:2007.04023*, 2020.

[21] Muhammad Ghifary, W. Bastiaan Kleijn, Mengjie Zhang, David Balduzzi, and Wen Li. Deep Reconstruction-Classification Networks for Unsupervised Domain Adaptation. In *Proceedings of European Conference on Computer Vision (ECCV)*, Lecture Notes in Computer Science, pages 597–613, 2016.

[22] Kotaro Hara, Abigail Adams, Kristy Milland, Saiph Savage, Chris Callison-Burch, and Jeffrey P. Bigham. A Data-Driven Analysis of Workers' Earnings on Amazon Mechanical Turk. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems*, pages 1–14, 2018.

[23] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep Residual Learning for Image Recognition. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778, 2016.

[24] Sungeun Hong, Woobin Im, Jongbin Ryu, and Hyun S. Yang. SSPP-DAN: Deep domain adaptation network for face recognition with single sample per person. In *Proceedings of IEEE International Conference on Image Processing (ICIP)*, pages 825–829, 2017.

[25] Matthew Hutson. Artificial intelligence faces reproducibility crisis. *Science*, 359: 725–726, 2018.

[26] Amazon Mechanical Turk Inc. Amazon Mechanical Turk. https://www.mturk.com/worker, 2018. Accessed: 2021-06-21.

[27] Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A. Efros. Image-to-Image Translation with Conditional Adversarial Networks. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5967–5976, 2017.

[28] Diederik Kingma and Jimmy Ba. Adam: A Method for Stochastic Optimization. In *Proceedings of International Conference on Learning Representations (ICLR)*, 2014.

[29] Ming-Yu Liu and Oncel Tuzel. Coupled Generative Adversarial Networks. *Advances in Neural Information Processing Systems*, 29, 2016.

[30] Mingsheng Long, Han Zhu, Jianmin Wang, and Michael I. Jordan. Deep transfer learning with joint adaptation networks. In *Proceedings of International Conference on Machine Learning (ICML)*, pages 2208–2217, 2017.

[31] Javier Marin, David Vazquez, David Geronimo, and Antonio M. Lopez. Learning appearance in virtual scenarios for pedestrian detection. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 137–144, 2010.

[32] Microsoft. AI for Earth Grants - Microsoft AI. https://www.microsoft.com/en-us/ai/ai-for-earth-grants, 2021. Accessed: 2021-06-21.

[33] Mohammad Sadegh Norouzzadeh, Anh Nguyen, Margaret Kosmala, Alexandra Swanson, Meredith S. Palmer, Craig Packer, and Jeff Clune. Automatically identifying, counting, and describing wild animals in camera-trap images with deep learning. *Proceedings of the National Academy of Sciences*, 115(25):E5716–E5725, 2018.

[34] S. J. Pan and Q. Yang. A Survey on Transfer Learning. *IEEE Transactions on Knowledge and Data Engineering*, 22(10):1345–1359, 2010.

[35] Xingchao Peng and Kate Saenko. Synthetic to Real Adaptation with Generative Correlation Alignment Networks. In *2018 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pages 1982–1991, 2018.

[36] L. Pishchulin, A. Jain, M. Andriluka, T. Thormahlen, and B. Schiele. Articulated people detection and pose estimation: Reshaping the future. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3178–3185, 2012.

[37] Leonid Pishchulin, Arjun Jain, Christian Wojek, Mykhaylo Andriluka, Thorsten Thormahlen, and Bernt Schiele. Learning people detection models from few training samples. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1473–1480, 2011.

[38] Ruslan Salakhutdinov, Antonio Torralba, and Josh Tenenbaum. Learning to share visual appearance for multiclass object detection. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1481–1488, 2011.

[39] Swami Sankaranarayanan, Yogesh Balaji, Arpit Jain, Ser Nam Lim, and Rama Chellappa. Learning From Synthetic Data: Addressing Domain Shift for Semantic Segmentation. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3752–3761, 2018.

[40] Baochen Sun and Kate Saenko. Deep CORAL: Correlation Alignment for Deep Domain Adaptation. In *Workshops of European Conference on Computer Vision (ECCV)*, Lecture Notes in Computer Science, pages 443–450, 2016.

[41] Baochen Sun, Jiashi Feng, and Kate Saenko. Return of Frustratingly Easy Domain Adaptation. In *Proceedings of AAAI Conference on Artificial Intelligence*, volume 30, 2016.

[42] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. Rethinking the Inception Architecture for Computer Vision. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2818–2826, 2016.

[43] Eric Tzeng, Judy Hoffman, Ning Zhang, Kate Saenko, and Trevor Darrell. Deep Domain Confusion: Maximizing for Domain Invariance. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2014.

[44] Eric Tzeng, Judy Hoffman, Kate Saenko, and Trevor Darrell. Adversarial Discriminative Domain Adaptation. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 7167–7176, 2017.

[45] Laurens van der Maaten and Geoffrey Hinton. Visualizing Data using t-SNE. *Journal of Machine Learning Research*, 9(86):2579–2605, 2008.

[46] Grant Van Horn and Pietro Perona. The Devil is in the Tails: Fine-grained Classification in the Wild. *arXiv:1709.01450*, 2017.

[47] Gül Varol, Javier Romero, Xavier Martin, Naureen Mahmood, Michael J. Black, Ivan Laptev, and Cordelia Schmid. Learning from Synthetic Humans. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4627–4635, 2017.

[48] Mei Wang and Weihong Deng. Deep visual domain adaptation: A survey. *Neurocomputing*, 312:135–153, 2018.

[49] Qi Wang, Junyu Gao, Wei Lin, and Yuan Yuan. Learning From Synthetic Data for Crowd Counting in the Wild. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 8198–8207, 2019.

[50] Hongliang Yan, Yukang Ding, Peihua Li, Qilong Wang, Yong Xu, and Wangmeng Zuo. Mind the Class Weight Bias: Weighted Maximum Mean Discrepancy for Unsupervised Domain Adaptation. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 945–954, 2017.

[51] Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A. Efros. Unpaired Image-to-Image Translation Using Cycle-Consistent Adversarial Networks. In *Proceedings of IEEE International Conference on Computer Vision (ICCV)*, pages 2242–2251, 2017.

[52] Xiangxin Zhu, Dragomir Anguelov, and Deva Ramanan. Capturing Long-Tail Distributions of Object Subcategories. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 915–922, 2014.

[53] Fuzhen Zhuang, Xiaohu Cheng, Ping Luo, Sinno Jialin Pan, and Qing He. Supervised representation learning: Transfer learning with deep autoencoders. In *Proceedings of International Joint Conference on Artificial Intelligence (IJCAI)*, pages 4119–4125, 2015.