# Augmenting Image Classifiers using Data Augmentation Generative Adversarial Networks

Antreas Antoniou[1], Amos Storkey[1], and Harrison Edwards[1,2]

[1] University of Edinburgh, Edinburgh, UK
{a.antoniou,a.storkey,h.l.edwards}@sms.ed.ac.uk
https://www.ed.ac.uk/
[2] Open AI
https://openai.com/

**Abstract.** Effective training of neural networks requires much data. In the low-data regime, parameters are underdetermined, and learnt networks generalise poorly. Data Augmentation alleviates this by using existing data more effectively, but standard data augmentation produces only limited plausible alternative data. Given the potential to generate a much broader set of augmentations, we design and train a generative model to do data augmentation. The model, based on image conditional Generative Adversarial Networks, uses data from a source domain and learns to take a data item and augment it by generating other within-class data items. As this generative process does not depend on the classes themselves, it can be applied to novel unseen classes. We demonstrate that a Data Augmentation Generative Adversarial Network (DAGAN) augments classifiers well on Omniglot, EMNIST and VGG-Face.

## 1 Introduction

Over the last decade Deep Neural Networks have enabled unprecedented performance on a number of tasks. They have been demonstrated in many domains [12] including image classification [25, 17, 16, 18, 21], machine translation [44], natural language processing [12], speech recognition [19], and synthesis [42], learning from human play [6] and reinforcement learning [27, 35, 10, 40, 13] among others. In all cases, very large datasets have been utilized, or in the case of reinforcement learning, extensive play. In many realistic settings we need to achieve goals with limited datasets; in those cases deep neural networks seem to fall short, overfitting on the training set and producing poor generalisation on the test set.

Techniques have been developed over the years to help combat overfitting such as L1/L2 reqularization [28], dropout [20], batch normalization [23], batch renormalisation [22] or layer normalization [2]. However in low data regimes, even these techniques fall short, since the the flexibility of the network is so high. These methods are not able to capitalise on known input invariances that might form good prior knowledge for informing the parameter learning.

It is also possible to generate more data from existing data by applying various transformations [25] to the original dataset. These transformations include random translations, rotations and flips as well as addition of Gaussian noise. Such methods capitalize on transformations that we know should not affect the class. This technique seems to be vital, not only for the low-data cases but for any size of dataset, in fact even models trained on some of the largest datasets such as Imagenet [7] can benefit from this practice.

Typical data augmentation techniques use a limited set of known invariances that are easy to invoke. Here, we recognize that we can learn a model of a much larger invariance space through training a form of conditional generative adversarial network (GAN) in some *source domain*. This can then be applied in the low-data domain of interest, the *target domain*. We show that such a Data Augmentation Generative Adversarial Network (DAGAN) enables effective neural network training even in low-data target domains. As the DAGAN does not depend on the classes themselves it captures the cross-class transformations, moving data-points to other points of equivalent class.

In this paper we train a DAGAN and then evaluate its performance on low-data tasks using standard stochastic gradient descent neural network training. We use 3 datasets, the Omniglot dataset, the EMNIST dataset and the more complex VGG-Face dataset. The DAGAN trained on Omniglot was used for augmenting both the Omniglot and EMNIST classifiers to demonstrate benefit even when transferring between substantially different domains. The VGG-Face dataset provides a considerably more challenging test for the DAGAN. VGG-Face was used to evaluate whether the DAGAN training scheme could work on human faces, which are notoriously hard to model using a generator. Furthermore the usefulness of the generated faces was measured when used as augmentation data in the classification training.

## 2 Background

**Transfer Learning and Dataset Shift**: The term *dataset shift* [36] generalises the concept of covariate shift [33, 37, 38] to multiple cases of changes between domains. For data augmentation, we may learn a generative distribution that maintains class consistency on one set of classes and apply that consistency transformation to new unseen classes, on the understanding the the transformations that maintain consistency generalise across classes.

**Generative Adversarial Networks**: GANs [11], and specifically Deep Convolutional GANs (DCGAN) [29] use the ability to discriminate between true and generated examples as a learning objective for generative models. GAN approaches can learn complex joint densities. Recent improvements in the optimization process [1, 14, 3] have reduced some of the failure modes of the GAN learning process as well as produced objectives that correlate well with sample quality [1, 14]. Furthermore image conditional GANs have been used to achieve image to image translation [24], as well as augment datasets [5, 45, 34]. However the work relating to the enhancement of datasets only uses the GAN to either fine tune simulated data or generate data by attempting to reconstruct existing data points. Whereas our model is explicitly trained to produce data augmentations as a manifold of samples around real data samples.

As demonstrated in [14], the Wasserstein formulation for training GANs has shown superior sample diversity and quality in multiple instances. Additionally the Wasserstein GANs (WGAN) with Gradient Penalty (GP) have the additional benefit of being trainable using advanced architectures such as ResNets [16]. This is especially important since most GAN formulations can only be successfully trained using very specific and often less expressive model architectures. Furthermore WGAN with GP discriminator losses have been empirically observed to correlate with sample quality. Taking into consideration available state of the art methods including standard GAN, LS-GAN, WGAN with clipping and WGAN with Spectral normalization, we focus on the use WGAN with GP training in this paper due to its versatility in terms of architectures and its superior qualitative performance. Our own experiments with other approaches confirm the stated benefits; we found WGAN with GP to produce the most stable models with the best sample quality both qualitatively and quantitatively.

**Data Augmentation**: Data augmentation similar to [25] is routinely used in classification problems. Often it is non-trivial to encode known invariances in a model. It can be easier to encode those invariances in the data instead by generating additional data items through transformations from existing data items. For example the labels of handwritten characters should be invariant to small shifts in location, small rotations or shears, changes in intensity, changes in stroke thickness, changes in size etc. Almost all cases of data augmentation are from a priori known invariance. Various attempts at augmenting *features* instead of data are investigated in [39, 8]. Moreover, the effectiveness of data augmentation has also been shown in other domains except images. Two such domains is sound [32] and text [31]. There has been little previous work that attempts to learn data augmentation strategies. One paper that is worthy of note is the work of [15], where the authors learn augmentation strategies on a class by class basis. Additional papers that attempt to learn models for data augmentation include [9, 4, 30]. These approaches do not transfer to the setting where completely new classes are considered.

## 3 Model

If we know that a class label should be invariant to a particular transformation then we can apply that transformation to generate additional data. If we do not know what transformations might be valid, but we have other data from related problems, we can attempt to learn valid transformations from those related problems that we can apply to our setting. This is an example of meta-learning; we learn on other problems how to improve learning for our target problem.

### 3.1 Model Overview

Consider a collection of datasets $[(x_i^c|i = 1, 2, \dots N^c)|c \in C]$, with each dataset labelled by $c$, the class, taken from the set of classes $C$, and with each element in a dataset $c$ indexed by $i$ and denoted by $x_i^c$. Let $x_i^c \in X$, the space of inputs. In this paper $X$ will be a space of input images.

The goal is to learn a mapping between a conditional sample $x_i^c$ of a certain class $c$ to other samples $x_j^c$ from that same class, using training data $[(x_i^c|i = 1, 2, \ldots N^c)|c \in C]$. To do so we learn a differentiable function $G$ which we call a generator. Given some random standard Gaussian vector $z$, we require a mapping $G : (x_i^c, z)$ such that, $\forall j$, $x_j^c$ has high probability under the density of $z$ mapped through G. Since G is differentiable, $z$ maps out a whole manifold in $X$ space associated with input $x_i^c$ in a class consistent way. Yet $G$ does not have access to the class $c$ itself, thus enabling the DAGAN to generalize to unseen classes. We parameterize our generator function $\tilde{x} = G(x_i^c, z)$ as a neural network and we train it as a GAN using the WGAN with GP formulation. Training a GAN also requires a discriminator network, denoted as $D$, to be trained along with the generator network. The discriminator network attempts to discriminate between real and fake samples whilst the generator attempts to minimize the discriminator's performance in guessing real from fake.

## 3.2 Model Objective Definition

We modify the WGAN with GP formulation to account for the fact that we are using an image-conditional GAN with a discriminator that takes as input 2 images, instead of 1. Figure 1 shows the high level overview of our training setup. Our generator and discriminator objectives can be expressed as:

$$L_{discr} = \underset{\tilde{x} \sim P_g}{\mathbb{E}}[D(x_i^c, \tilde{x})] - \underset{X \sim P_r}{\mathbb{E}}[D(x_i^c, x_j^c)] + \lambda \underset{\hat{x} \sim P_{\hat{x}}}{\mathbb{E}}(||\nabla_{\hat{x}} D(x_i^c, \hat{x})||_2 - 1) \quad (1)$$

$$L_{gen} = - \underset{\tilde{x} \sim P_g}{\mathbb{E}}[D(x_i^c, \tilde{x})], \quad (2)$$

where $x$ represents real samples, $x_i^c$ and $x_j^c$ represent two separate instances of samples from class c, $\tilde{x}$ represents generated samples from the generator G. $\hat{x}$ is, as defined in [14], randomly sampled points on linear interpolations between the samples of the real distribution $P_r$ and generated distribution $P_g$. The only difference from the original WGAN with GP formulation is the use of 2 entries in the discriminator arguments, one for the conditional sample $x_i^c$ and one for the target sample $x_j^c$ (for real case) or $\tilde{x}$ (for fake case).

## 3.3 Architectures

We chose to use a state of the art Densenet discriminator and, for the generator, a powerful combination of two standard networks, UNet and ResNet, which we henceforth call a UResNet. The code for this paper is available[3], and that provides the full implementation of the networks. However we describe the implementational details here.

[3] https://github.com/AntreasAntoniou/DAGAN

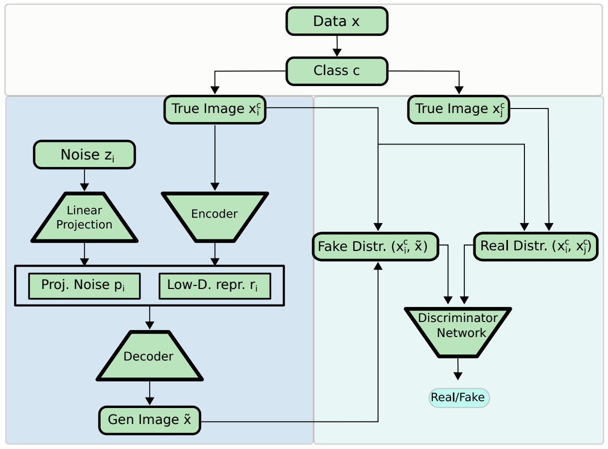

**Fig. 1.** DAGAN Architecture. Left: the generator network is composed of an encoder taking an input image and projecting it to a lower dimensional manifold. A random vector ($z$) is transformed and concatenated with the bottleneck vector; these are both passed to the decoder network which generates a within-class image. Right: the adversarial discriminator network is trained to discriminate between the samples from the *real* distribution (two real images from the same class) and the *fake* distribution (a real sample and a generated sample). Adversarial training enables the network to generate within-class images that look different enough to be considered a different sample.

The UResNet generator has a total of 8 blocks, each block having 4 convolutional layers (with leaky rectified linear (ReLU) activations and batch renormalisation (batchrenorm) [22]) followed by one downscaling or upscaling layer. Downscaling layers (in blocks 1-4) were convolutions with stride 2 followed by leaky ReLU, batch normalisation and dropout. Upscaling layers were implemented by employing a nearest neighbour upscale, followed by a convolution, leaky ReLU, batch renormalisation and dropout. For Omniglot and EMNIST experiments, all layers had 64 filters. For the VGG-Face experiments the first 2 blocks of the encoder and the last 2 blocks of the decoder had 64 filters and the last 2 blocks of the encoder and the first 2 blocks of the decoder 128 filters.

In addition each block of the UResNet generator had skip connections. As with a standard ResNet, we used either a summation skip connection between layers with equivalent spacial dimensions or a strided 1x1 convolution for between layers with different spacial dimensions, thus bypassing the between block non-linearity to help gradient flow. Finally skip connections were introduced between equivalent sized filters at each end of the network (as with UNet).

We used a DenseNet [21] discriminator, using layer normalization instead of batch normalization; the latter would break the assumptions of the WGAN objective function (as mentioned in [[14] chapter 4). The DenseNet was composed of 4 Dense Blocks and 4 Transition Layers, as defined in [21]. We used a growth rate of $k = 64$ and each Dense Block had 5 convolutional layers. We removed the 1x1 convolutions usually before the 3x3 convolutions as we observed this improved sample quality. For the discriminator we used dropout at the last convolutional layer of each Dense Block; this too improved sample quality.

For each classification experiment we used a DenseNet classifier composed of 4 Dense Blocks and 4 Transition Layers with a $k = 64$, each Dense Block had 3 convolutional layers within it. The classifiers were a total of 17 layers (i.e. 16 layers and 1 softmax layer). Furthermore we applied a dropout of 0.5 on the last convolutional layer in each Dense Block.

## 4 Datasets and Experiments

We tested the DAGAN augmentation on 3 datasets: Omniglot, EMNIST, and VGG-Face. All datasets were split randomly into source domain sets, validation domain sets and test domain sets.

For classifier networks, data for each character (handwritten or person) was further split into 2 test cases (for all datasets), 3 validation cases and a varying number of training cases depending on the experiment. Classifier training was done on the training cases for all examples in all domains; hyperparameter choice used validation cases. Test performance was reported only on the test cases for the target domain set. Case splits were randomized across each test run.

The Omniglot data [26] was split into source domain and target domain similarly to the split in [41]. The class ids were sorted in an increasing manner. The first 1200 were used as a source domain set, 1201-1412 as a validation domain set and 1412-1623 as a target domain test set.

The EMNIST data was split into a source domain that included classes 0-34 (after random shuffling of the classes), the validation domain set included classes 35-42 and the test domain set included classes 42-47. Since the EMNIST dataset has thousands of samples per class we chose only a subset of 100 for each class, so that we could make our task a low-data one.

In the VGG-Face dataset case, we randomly chose 100 samples from each class that had 100 or more, uncorrupted images, resulting in 2396 of the full 2622 classes available in the dataset. After shuffling, we split the resulting dataset into a source domain that included the first 1802 classes. The test domain set included classes 1803-2300 and the validation domain set included classes 2300-2396.

### 4.1 Training of DAGAN in source domain

A DAGAN was trained on Source Omniglot domains using a variety of architectures: standard VGG, U-Net, and ResNet inspired architectures. Increasingly powerful networks proved better generators, with the UResNet described in Section 3.3 generator being our model of choice. Examples of generated data are given in Figure 2. We trained each DAGAN for 200K iterations, using a learning rate of 0.0001, and an Adam optimizer with Adam parameters of $\beta_1 = 0$ and $\beta_2 = 0.9$. We used a pretrained DenseNet classifier to quantify the performance of the generated data in terms of how well they classify in real classes. We chose the model that had the best validation accuracy performance on this classifier.

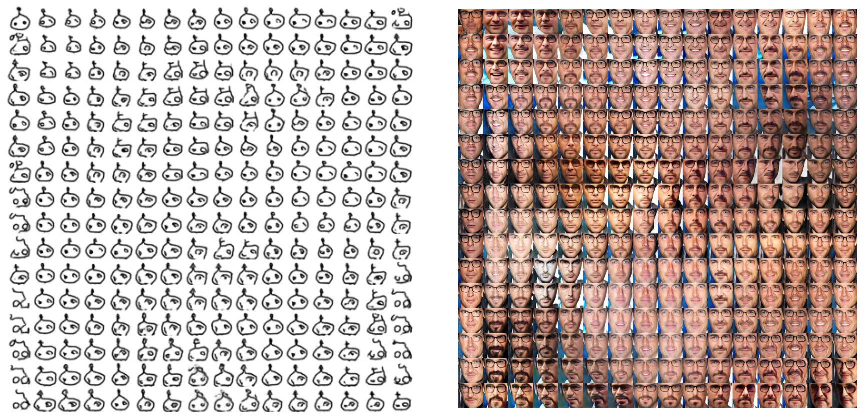

**Fig. 2.** An Interpolated spherical subspace[43] of the GAN generation space on Omniglot and VGG-Face respectively. The only real image ($x_i^c$) in each figure is the one in the top-left corner, the rest are generated to augment that example using a DAGAN.

.

### 4.2 Classifiers

The primary question of this paper is how well the DAGAN can augment vanilla classifiers trained on each target domain. A DenseNet classifier (as described in Section 3.3) was trained first on just real data (with standard data augmentation) with 5 to 100 examples per class (depending on dataset). In the second case, the classifier was was also trained on DAGAN generated data. The real or fake label was also passed to the network, via adding 1 filter before each convolution of either zeros (fake) or ones (real) to enable the network to learn how best to emphasise true over generated data. This last step proved crucial to maximizing the potential of the DAGAN augmentations. In each training cycle, varying numbers of augmented samples were provided for each real example (ranging from 1-10). The best hyperparameters were selected via performance on the validation domain. The classifier was trained with standard augmentation: random Gaussian noise was added to images (with 50% probability), random shifts along x and y axis (with 50% probability), and random 90 degree rotations (all with equal probability of being chosen). Classifiers were trained for 200 epochs, with a learning rate of 0.001, and an Adam optimizer with $\beta_1 = 0.9$ and $\beta_2 = 0.99$. The results on the held out test cases from the target domain is given in Table 1. In every case the augmentation improves the classification.

## 5 Conclusions

Data augmentation is a widely applicable approach to improving performance in low-data settings. The DAGAN is a flexible model to automatically learn to augment data. We demonstrate that a DAGAN can improve performance of classifiers even after standard data-augmentation. Furthermore, it is worth

| Samples Per Class | Augment with DAGAN | Omniglot | EMNIST | VGG-Face |
|---|---|---|---|---|
| 5 | False | 68.99% | - | 04.47% |
| 5 | True | **82.13%** | - | **12.59%** |
| 10 | False | 79.41% | - | - |
| 10 | True | **86.22%** | - | - |
| 15 | False | 81.97% | 73.93% | 39.33% |
| 15 | True | **87.42%** | **76.07%** | **42.93%** |
| 25 | False | - | 78.35% | 57.99% |
| 25 | True | - | **80.26%** | **58.46%** |
| 50 | False | - | 81.51% | - |
| 50 | True | - | **82.78%** | - |
| 100 | False | - | 83.78% | - |
| 100 | True | - | **84.80%** | - |

**Table 1.** Classification Results: All results are averages over 5 independent runs. The DAGAN augmentation improves the classifier performance in all cases. Test accuracy is the result on the test cases in the test domain. Here for the purposes of compactness we omit the *number of generated samples per real sample* hyperparameter since that would produce more than 100 rows of data. We should note however that the optimal number of generated samples per real image was found to be 3.

noting that a DAGAN can be easily combined with other model types, including few shot learning models. Further work is needed to evaluate the usefulness in the few shot learning. However the flexibility of the DAGAN makes it a powerful means of enhancing models working with a small amount of data.

***Acknowledgements*** This work was supported in by the EPSRC Centre for Doctoral Training in Data Science, funded by the UK Engineering and Physical Sciences Research Council and the University of Edinburgh as well as by the European Unions Horizon 2020 research and innovation programme under grant agreement No 732204 (Bonseyes) and by the Swiss State Secretariat for Education Research and Innovation (SERI) under contract number 16.0159. The opinions expressed and arguments employed herein do not necessarily reflect the official views of these funding bodies.

## References

1. Arjovsky, M., Chintala, S., Bottou, L.: Wasserstein gan. arXiv:1701.07875 (2017)
2. Ba, J.L., Kiros, J.R., Hinton, G.E.: Layer normalization. arXiv:1607.06450 (2016)
3. Berthelot, D., Schumm, T., Metz, L.: Began: Boundary equilibrium generative adversarial networks. arXiv:1703.10717 (2017)
4. Bloice, M.D., Stocker, C., Holzinger, A.: Augmentor: An image augmentation library for machine learning. arXiv:1708.04680 (2017)
5. Choe, J., Park, S., Kim, K., Park, J.H., Kim, D., Shim, H.: Face generation for low-shot learning using generative adversarial networks. In: Computer Vision Workshop (ICCVW), 2017 IEEE International Conference on. IEEE (2017)
6. Clark, C., Storkey, A.: Training deep convolutional networks to play Go. In: Proceedings of 32nd International Conference on Machine Learning (ICML2015) (2015), (arxiv 2014)

7. Deng, J., Dong, W., Socher, R., Li, L.J., Li, K., Fei-Fei, L.: Imagenet: A large-scale hierarchical image database. In: Computer Vision and Pattern Recognition. IEEE (2009)
8. Dixit, M., Kwitt, R., Niethammer, M., Vasconcelos, N.: Aga: Attribute-guided augmentation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (2017)
9. Fawzi, A., Samulowitz, H., Turaga, D., Frossard, P.: Adaptive data augmentation for image classification. In: Internationa Conference on Image Processing (ICIP), 2016. IEEE (2016)
10. Foerster, J., Assael, Y.M., de Freitas, N., Whiteson, S.: Learning to communicate with deep multi-agent reinforcement learning (2016)
11. Goodfellow, I.J., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., Bengio, Y.: Generative Adversarial Networks (June 2014)
12. Gu, J., Wang, Z., Kuen, J., Ma, L., Shahroudy, A., Shuai, B., Liu, T., Wang, X., Wang, G.: Recent Advances in Convolutional Neural Networks (2015)
13. Gu, S., Lillicrap, T., Sutskever, I., Levine, S.: Continuous deep q-learning with model-based acceleration. In: International Conference on Machine Learning (2016)
14. Gulrajani, I., Ahmed, F., Arjovsky, M., Dumoulin, V., Courville, A.: Improved training of wasserstein gans. arXiv:1704.00028 (2017)
15. Hauberg, S., Freifeld, O., Larsen, A.B.L., Fisher, J., Hansen, L.: Dreaming more data: Class-dependent distributions over diffeomorphisms for learned data augmentation. In: Artificial Intelligence and Statistics (2016)
16. He, K., Zhang, X., Ren, S., Sun, J.: Deep Residual Learning for Image Recognition (dec 2015)
17. He, K., Zhang, X., Ren, S., Sun, J.: Delving Deep into Rectifiers: Surpassing Human-Level Performance on ImageNet Classification (2015)
18. He, K., Zhang, X., Ren, S., Sun, J.: Identity mappings in deep residual networks. In: European Conference on Computer Vision. Springer (2016)
19. Hinton, G., Deng, L., Yu, D., Dahl, G.E., Mohamed, A.r., Jaitly, N., Senior, A., Vanhoucke, V., Nguyen, P., Sainath, T.N., et al.: Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups. IEEE Signal Processing Magazine (2012)
20. Hinton, G.E., Srivastava, N., Krizhevsky, A., Sutskever, I., Salakhutdinov, R.R.: Improving neural networks by preventing co-adaptation of feature detectors (2012)
21. Huang, G., Liu, Z., Weinberger, K.Q., van der Maaten, L.: Densely connected convolutional networks. arXiv:1608.06993 (2016)
22. Ioffe, S.: Batch renormalization: Towards reducing minibatch dependence in batch-normalized models (2017)
23. Ioffe, S., Szegedy, C.: Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift (2015)
24. Isola, P., Zhu, J.Y., Zhou, T., Efros, A.A.: Image-to-Image Translation with Conditional Adversarial Networks (2016)
25. Krizhevsky, A., Sutskever, I., Hinton, G.E.: Imagenet classification with deep convolutional neural networks. In: Advances in neural information processing systems (2012)
26. Lake, B.M., Salakhutdinov, R., Tenenbaum, J.B.: Human-level concept learning through probabilistic program induction. Science (2015)
27. Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A.A., Veness, J., Bellemare, M.G., Graves, A., Riedmiller, M., Fidjeland, A.K., Ostrovski, G., et al.: Human-level control through deep reinforcement learning. Nature (2015)

28. Nowlan, S.J., Hinton, G.E.: Simplifying neural networks by soft weight-sharing. Neural Computation (1992). https://doi.org/10.1162/neco.1992.4.4.473
29. Radford, A., Metz, L., Chintala, S.: Unsupervised representation learning with deep convolutional generative adversarial networks. In: Proceedings of ICLR 2016 (2015)
30. Ratner, A.J., Ehrenberg, H., Hussain, Z., Dunnmon, J., Ré, C.: Learning to compose domain-specific transformations for data augmentation. In: Advances in Neural Information Processing Systems 30 (2017)
31. Rosn, B.: Asymptotic theory for order sampling. Journal of Statistical Planning and Inference (1997)
32. Salamon, J., Bello, J.P.: Deep convolutional neural networks and data augmentation for environmental sound classification. IEEE Signal Processing Letters (2017)
33. Shimodaira, H.: Improving predictive inference under covariate shift by weighting the log-likelihood function. Journal of Statistical Planning and Inference **90**, 227–244 (2000)
34. Shrivastava, A., Pfister, T., Tuzel, O., Susskind, J., Wang, W., Webb, R.: Learning from simulated and unsupervised images through adversarial training. In: The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2017)
35. Silver, D., Huang, A., Maddison, C.J., Guez, A., Sifre, L., Van Den Driessche, G., Schrittwieser, J., Antonoglou, I., Panneershelvam, V., Lanctot, M., et al.: Mastering the game of go with deep neural networks and tree search. Nature (2016)
36. Storkey, A.: When training and test sets are different: Characterising learning transfer. In: Lawrence, C.S.S. (ed.) Dataset Shift in Machine Learning, chap. 1. MIT Press (2009)
37. Storkey, A., Sugiyama, M.: Mixture regression for covariate shift. In: Advances in Neural Information Processing Systems 19 (NIPS2006) (2007)
38. Sugiyama, M., Müller, K.R.: Input-dependent estimation of generalisation error under covariate shift. Statistics and Decisions (2005)
39. Takeki, A., Ikami, D., Irie, G., Aizawa, K.: Parallel grid pooling for data augmentation. arXiv:1803.11370 (2018)
40. Van Hasselt, H., Guez, A., Silver, D.: Deep reinforcement learning with double q-learning. In: AAAI (2016)
41. Vinyals, O., Blundell, C., Lillicrap, T., Wierstra, D., et al.: Matching networks for one shot learning. In: Advances in Neural Information Processing Systems (2016)
42. Wang, Y., Skerry-Ryan, R.J., Stanton, D., Wu, Y., Weiss, R.J., Jaitly, N., Yang, Z., Xiao, Y., Chen, Z., Bengio, S., Le, Q.V., Agiomyrgiannakis, Y., Clark, R., Saurous, R.A.: Tacotron: A fully end-to-end text-to-speech synthesis model. CoRR **abs/1703.10135** (2017)
43. White, T.: Sampling Generative Networks (September 2016)
44. Wu, Y., Schuster, M., Chen, Z., Le, Q.V., Norouzi, M., Macherey, W., Krikun, M., Cao, Y., Gao, Q., Macherey, K., et al.: Google's neural machine translation system: Bridging the gap between human and machine translation. arXiv:1609.08144 (2016)
45. Xian, Y., Lorenz, T., Schiele, B., Akata, Z.: Feature generating networks for zero-shot learning. arXiv preprint arXiv:1712.00981 (2017)