IMPROVING DISCRIMINATOR-GENERATOR BALANCE IN GENERATIVE ADVERSARIAL NETWORKS

Anonymous authors
Paper under double-blind review

ABSTRACT

Training Artificial Neural Networks to generate better images is a hard problem to solve with supervised techniques. Generative Adversarial Networks (GAN) is an unsupervised approach that utilizes two ANNs in one system. The first neural network is called the Discriminator and evaluates the quality of images generated by the other network, the Generator. However, image generation has no fixed solution, making evaluation difficult. This can be addressed through comparing two GAN models by letting them evaluate each other; the paper proposes a custom implementation of this evaluation method that allows for comparison between two GAN models both overall and throughout the learning process.

GANs have a reputation for being hard to train. One concrete problem is maintaining the balance between the Generator and Discriminator. As for humans, it is easier to rate the quality of images then it is to actually create them. A good evaluator is necessary, but it must not out-power the generative model. The paper explores two main approaches to achieve a better balanced GAN model. The first method makes guided alterations to the usually random input of the Generator. The second method adds an additional Discriminator to the model. Techniques based on both of these methods were shown to effectively guide the training process and creating strong models that outperformed regular GANs when compared using the previously mentioned evaluation method.

1 INTRODUCTION

Images generated by Artificial Neural Networks (ANNs) are often vivid and impressive, but do not appear natural to a human viewer. This is not surprising as the utilized networks are usually trained to maximise object recognition, as with Mordvintsev et al. (2015) demonstrating how vivid imagery could be generated by Google’s Inception network, normally used to classify objects in images. The common approach to ANNs is to use a single network and train it with labelled training data in a supervised fashion. Training a network to generate more natural looking images is certainly possible, but would be a hard problem to solve with conventional supervised learning, requiring either a human to evaluate every generated image or creating a large dataset with examples of good and bad images. Nightmare Machine[1] uses a version of the former, allowing online users to rate generated samples as either scary or not scary.

The innovative approach with Generative Adversarial Networks (GAN) is to replace the human evaluator with an additional neural network (Goodfellow et al. [2014]). The first network, the generative model, is tasked with generating new samples, often images, based on an existing set of samples. The second network, the discriminative model, evaluates and judges whether the sample is from the dataset or synthesized by the other model. The two networks partake in an adversarial zero sum min-max battle to improve their own performance, that is, the Discriminator’s ability to separate real samples from the ones generated by the Generator vs. the Generator’s capacity to generate samples of improved quality to deceive the Discriminator. The result of this battle can be a model capable of generating new natural looking images based on unsupervised learning of an image dataset.

More formally the GAN’s min-max game can be described by the following function $V$:

$$ V(D, G) = \mathbb{E}_{x \sim p_{\text{data}}(x)} \left[ \log D(x) \right] + \mathbb{E}_{z \sim p_{\text{noise}}(z)} \left[ \log (1 - D(G(z))) \right] $$

where $x$ is a sample from the training dataset which has the probability distribution $p_{data}$; $z$ is a randomly generated noise sample from the distribution $p_{noise}$ (often a list of random numbers in range -1 to 1); $E_x$ and $E_z$ indicate which data distribution the sample belongs to: $E_x$ denotes real samples and $E_z$ denotes generated ones; $G(z)$ represents a sample generated by the Generator model $G$ with noise $z$ as input; and $D(x)$ is the probability that the sample $x$ is a real sample and not generated by $G$.

GANs have evolved considerably since their inception, but there are still open problems. One is how to objectively evaluate a GAN model. Image generation has no fixed solution, and evaluating it is as difficult as evaluating other creative works. One method allows for two GAN models to be compared by evaluating each other. This method leverages the adversarial min-max aspect of GANs, by having the Discriminator from one model classify generated samples from the other, and vice versa. This paper introduces a custom implementation of this method that allows for comparison between two GAN models both overall and throughout the learning process.

Another problem with GANs is in maintaining the balance between the Generator and Discriminator during training. As for humans, it is easier to rate the quality of images than it is to actually create them. As a result of this, the Discriminator’s performance is improving at a faster pace than the Generator’s. A good evaluator is necessary, but it must not out-power the generative model. The main method of addressing this problem has been to artificially limit the performance of the Discriminator. Goodfellow (2017) is critical to this approach as a well-trained Discriminator is required to accurately describe the probability density of the training data. Here two main approaches will be explored to achieve a better balanced GAN model. The first method makes guided alterations to the usually random input of the Generator. Three new techniques will be evaluated for altering the Generative model’s input to give it an edge during training.

The second method to balance the GAN adds an additional Discriminator to the model. The fundamental difference of GAN compared to usual ANN is the use of two networks in the same model. Adding additional neural networks is therefore a logical next step and a growing area of research. Although there are many ways of expanding the number of networks in a GAN, this work will focus on adding additional, asymmetric Discriminators only. Asymmetric does in this case entail that the Discriminators are different models and not just multiple random instantiations of the same design.

Techniques based on both of these methods were shown to effectively guide the training process and creating strong models that outperformed regular GANs when compared with the previously mentioned evaluation metric.

## 2 RELATED WORK

Several versions of Generative Adversarial Networks have been applied to image generation. Deep Convolutional Generative Adversarial Networks (DCGAN) (Radford et al., 2015) is an architecture that specializes GANs for image generation by applying advances from Convolutional Neural Networks. The generated images are far less noisy and much sharper with accurate colour representation than those generated by GAN. The architectural features of DCGAN also make the network flexible and able to dynamically adjust to most kinds of input. The same model that generated faces can be trained to generate handwritten digits and images of bedrooms without changing any parameters. [Radford et al., 2015] list three main techniques that DCGAN employs and that help improve the image generation: Batch normalization (Ioffe & Szegedy, 2015), the all convolutional network citepallcon, and sparse connectivity (Goodfellow et al., 2016).

Salimans et al. (2016) improve GANs further by presenting new architectural features that they apply to unsupervised and semi-supervised learning, in particular, Feature Matching which makes the Generator learn features directly from the training data (and not the Discriminator), Minibatch Discrimination which helps stabilize the Discriminator by letting it handle images in batches, and One-sided Label Smoothing which softens the outputs from the Discriminator to help balance the performance of the two models.

Arici & Celikyilmaz (2016) argue that many of the problems related to the training GAN, such as keeping the Discriminator and Generator synchronized are partly caused by the difficulty of mapping flat randomly generated noise, Generator input $Z$, and the training data. [Arici & Celikyilmaz] add an intermediate third neural network as link between the Discriminator and Generator. This additional
network, a Restricted Boltzmann machine (RBM), samples data from one of the intermediate hidden layers of the Discriminator to generate inputs to the Generator. These Associative Adversarial Networks replace the uniformly generated noise with samples more similar to training data. Arici & Celikyilmaz (2016) trained their system on CelebA and the third network was indeed able to learn the probabilistic model and generate samples based on it. The authors did not go into further detail about how it affected image generation, although they mentioned it did little to improve the learning gap between the Discriminator and Generator.

Durugkar et al. (2016) created a GAN system known as Generative Multi-Adversarial Network (GMAN) with multiple, symmetrical Discriminator models and a single Generator model. The Discriminators are instantiated with slightly varying parameters, but are otherwise architecturally the same and trained in similar fashion to regular GANs. The training of the Generator on the other hand is somewhat different. Each Discriminator evaluates and outputs its scores on the current generated sample. The scores are processed through a selection metric before being used to train the Generator.

Im et al. (2016b) argue that a single Generator/Discriminator pair is susceptible to not learning the whole distribution of the training data and suggest using multiple pairs as a way to address this problem. Generative Adversarial Parallelization (GAP) is the resulting system of this assumption. GAP removes the usual tight connection between Discriminator and Generator by randomly connecting pairs during training. Experiments show GAP to be robust against both mode coverage and mode collapse considering the Generators must generate images that fool multiple Discriminators. A single Discriminator can no longer overfit itself towards a single Generator, but must evolve to learn the characteristic of all Generators, which improves balance between models. As the names implies, all models are trained in parallel using multiple GPUs. Similarly, Durugkar et al. (2016) experimented with using ensembles of GAN, with multiple Discriminator and Generator pairs.

Shrivastava et al. (2016) change the Discriminator to addresses the instability of GAN and remove artefacts that are common in generated imagery, using local adversarial loss to divide the Discriminator field of operations into smaller local areas instead of the entire image, so that the Discriminator outputs multiple scores for each image compared to the regular single score. Shrivastava et al. (2016) use their GAN system to generate additional training data, resulting in a new dataset much larger than the original one. Using this dataset to train a regular CNN on gaze estimation reduced the error rate by 44%.

3 EVALUATION

One problem with GANs is that they have no obvious method of quantitatively evaluating the quality of the output. Assessing the quality of a generated image is hard, as there is neither a fixed answer nor a Boolean score. A generated image will seldom be completely unnatural nor natural, but have areas of both. Assessing the quality is also highly subjective. Theis et al. (2016) discussed some the more popular evaluation metrics and concluded that both the training and evaluation of a GAN should be modelled after the target application.

Using probabilistic based log-likelihood was originally proposed by Goodfellow et al. (2014) to compare the learned probability function of the training data. Goodfellow et al. did warn, however, that it was not suitable for data of higher dimensions, such as images. A GAN model can generate samples of high quality with poor likelihood and vice versa.

Nearest Neighbour algorithms can be used to evaluate GANs, e.g., by comparing the pixel data of generated samples to the training datasets. Durugkar et al. (2016) created a larger evaluation system where a separate CNN finds the generated images that are most similar to images in the training dataset. Ledig et al. (2016) used GAN to upscale the resolution of images with nearest neighbour as one of the measures of comparing the result to the training data. Theis et al. (2016) argues against using NN as an evaluation metric as even a small shift in the images can drastically change the distance.

The Visual Turing Test is an adaptation of the famous Turing test for human computer interactions. Salimans et al. (2016) used human annotators to assess the quality of their GAN. Users were shown images either generated or from the training set, and tasked with evaluating the images as either real or fake. Salimans et al. found this approach of evaluation less than desirable. Result were
dependent on the annotators and their motivation. Feedback on their own performance, such as which images were correctly classified or certain patterns to look for, made the annotators more successful. Experimentation with image generation using the CIFAR-10 dataset saw annotators correctly categorizing 78.7% of the images correctly as either real or fake. However, Salimans et al. themselves were able to get over 95% accuracy, as they were more familiar with CIFAR-10 and GAN.

Regular ANNs, and specifically CNNs, have become quite good at classifying objects in an image. These networks are trained on a large set of labelled images. When shown an image, the network will output which objects it believes is in the picture and its degree of confidence. These object classification networks can be used to evaluate the quality of the generated images from GAN. Salimans et al. (2016) created an evaluation system using the pre-trained Inception (Szegedy et al., 2016) model to classify objects in every generated sample and assuming generated images with clear objects are better. The model as a whole is evaluated by the number of different objects recognized, with variation in the generated samples preferred.

Generative Adversarial Metric (GAM) (Im et al., 2016a) is a method for comparing GANs. GAM leverages the adversarial min-max aspect of GANs, by having the Discriminator from one model classify generated samples from the other, and vice versa. The GAN with the least classification error may be the best model. Im et al. argue that the two Discriminators should also be verified against actual training data to ensure that one model is not overfitted more than the other. In other words, the two Discriminators must perform about equal on the test dataset for GAM to elect the better GAN. Equation 2 shows how the ratio of classification error for generated samples and test data are calculated. If \( r_{test} \approx 1 \), the value of \( r_{sample} \) is used to select a winning GAN.

\[
\begin{align*}
    r_{test} &= \frac{\epsilon(D_1(x))}{\epsilon(D_2(x))} \\
    r_{sample} &= \frac{\epsilon(D_1(G_2(z)))}{\epsilon(D_2(G_1(z)))}
\end{align*}
\]

where \( x \) is a sample from the training dataset which has the probability distribution \( p_{data} \), \( z \) is a randomly generated noise sample from the distribution \( p_{noise} \), \( G_y(z) \) represents a sample generated with noise \( z \) by Generator in GAN model \( y \), with \( y = \{1, 2\} \), \( D_y(x) \) is the probability that the sample \( x \) is a real sample and not generated, and \( \epsilon \) is the error rate of the Discriminator in the classification of images as real or generated.

Im et al. (2016b) and Durugkar et al. (2016) created variations of GAM to work with multiple Discriminators. Improvements in generalization resulted in the test ratio of Equation 2 failing when compared to other GANs. To solve this, Im et al. used either average error rate or selected the worst one across all the Discriminators.

4 Experimental Setup

Multiple open implementations of GAN were researched before settling on Taehoon Kim’s TensorFlow port of the Radford et al. (2015) original implementation of DCGAN. All experiments reported here took an extended version of Kim’s DCGAN system as starting point.

4.1 Data

Unsupervised learning is highly dependent on the training dataset as this is the source of all attained knowledge. Generally, datasets with higher degree of similarity between images result in higher quality. The number of images in the dataset is also important and larger datasets are preferred. Table I describes the selected datasets.

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CelebA</td>
<td>Large-scale Celeb Faces Attributes (Liu et al., 2015)</td>
</tr>
<tr>
<td>CIFAR-10</td>
<td>Collection of facial images of 11,117 celebrities</td>
</tr>
</tbody>
</table>

CifarA and CIFAR-10 are two publically available datasets. CelebA is available in two editions; one with images cropped and aligned and the other with original, in-the-wild images.
<table>
<thead>
<tr>
<th>Dataset</th>
<th>Size</th>
<th>Annotations</th>
<th>Resolution</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>CelebA</td>
<td>202,599</td>
<td>Name, description</td>
<td>Variable</td>
<td></td>
</tr>
<tr>
<td>CelebA Aligned</td>
<td>202,599</td>
<td>Name, description</td>
<td>178x218</td>
<td></td>
</tr>
<tr>
<td>CIFAR-10</td>
<td>60,000</td>
<td>Objects</td>
<td>32x32</td>
<td></td>
</tr>
<tr>
<td>IKEA Original</td>
<td>7,383</td>
<td>None</td>
<td>250x250</td>
<td></td>
</tr>
<tr>
<td>IKEA (Augmented)</td>
<td>191,828</td>
<td>None</td>
<td>250x250</td>
<td></td>
</tr>
</tbody>
</table>

Table 1: Summary of the selected datasets

CelebA has become quite popular for the use with GAN after [Radford et al.] used it with their DCGAN architecture. CIFAR-10 ([Krizhevsky & Hinton, 2009]) is a labelled subset of the Tiny Images Dataset ([Torralba et al., 2008]) and contains 60,000 images at a resolution of 32x32. The resolution is quite low and the content is often hard to make out.

The IKEA dataset was collected for this work and includes product images scraped from the furniture retailer IKEA’s website [http://www.ikea.com]. IKEA offers thousands of products with each product having images following the same guidelines. Every product image shares the same composition with the product aligned centrally and a white background. The data collection was time consuming and required much manual effort. The initial set contained some hundred thousand images with various depictions and dimensions. This set was reduced to about 10,000 by removing duplicates and images of resolution other than 250x250, but included ambiguous images showing whole rooms or product images for textile and other fabrics. These were removed by manual curation, with mostly furniture remaining. The final dataset contains about 7,000 images. The IKEA pictures share a common framing which makes them susceptible to augmentations, artificially increasing the size of the dataset by making slightly adjusted copies of the original images. The white background greatly expands possible alterations as the object can be moved around the image. The size of the dataset was artificially increased by implementing slight random transformations that include moving the products around, rotating, stretching and flipping the images.

4.2 Methodology

One of issues with GANs is the problem of synchronising the performance of the generative model and discriminative model during training. More specifically that the Discriminator may become too strong compared to the Generator. The Associative Adversarial Network proposed by [Arici & Celikyilmaz, 2016] attempted to balance the performance by making the learning task easier for the Generator. Their assumption was that the issue was partly caused by the difficulty of mapping flat noise samples to realistic output image. Their proposed solution was by using a third neural network as a link between the Generator and Discriminator. This approach is based on log-likelihood which multiple times has been discouraged with use on image data. Their approach also seems somewhat excessive and unnecessary since this associative network essentially acts as a second Generator. Still, their initial assumption is interesting and worth exploring further. Hence this paper will propose three techniques based on a similar assumption as the one by [Arici & Celikyilmaz, 2016], that the large distribution of the $Z$ noise is limiting the learning process of the Generator.

4.2.1 Static Reusable Noise

The discriminative model in a GAN is allowed to focus its learning on training dataset, while the Generator has the more difficult task of mapping a large, almost infinite set of random noise samples to natural generated images. Hence the Discriminator will learn on the same images every iteration while the Generator will seldom, if ever, see the same input during training. To balance this, in Static Reusable Noise (SRN) the noise samples are created once and then reused every epoch. The noise samples will therefore be familiar to Generator, which may better focus its effort at generating better images. The generated images will be quite similar each epoch and the feedback from the Discriminator will still be relevant during the next epoch. The noise samples are generated before training commences and are sampled through a seed set. This also enables the same noise to be reused during evaluation.
4.2.2 Image Based Noise Generation

In Image Based Noise Generation (IBNG) the noise samples created for the Generator are based on images from the training dataset. Samples from the training set are first scaled down to the smaller size of the noise data and then converted to black and white values in the range $[-1, 1]$, as illustrated in Figure 1. IBNG is similar to the Associative Network solution proposed by Arici & Celikyilmaz (2016) in that both approaches replace the regular noise with input more akin to the training data. IBNG is a simpler and less computationally expensive compared to the Associative Network.

4.2.3 Audition Based Noise Selection

Audition Based Noise Selections (ABNS) uses the Discriminator to select representative generated samples to guide the training. ABNS is a pre-processing technique that selects the noise to be used for training. It is based on an assumption that a strong Discriminator is able to accurately select generated samples that provide the most feedback to Generator. The technique is called audition based as it auditions a large set of noise samples where only a select few are chosen. The number of auditioned noise samples is set by the variable audition size which is multiplied with the batch size. For this paper, audition size is set to 3 and 6.

A selection metric picks images based on their score and assembles a regularly sized batch. This batch is used to train the system normally without any modifications to the GAN model. The selection metric has two modes:

- **Best** will only pick the images that attained the highest rating with the Discriminator.
- **Mixed** is used to give the Generator balanced feedback on its image generation. Mixed mode divides the batch in three and fills it with equal amounts of images with the highest and lowest rating. The rest are randomly generated.

4.2.4 Generative Multi-Adversarial Networks

Durugkar et al. (2016) used multiple Discriminators against a single Generator, while Im et al. (2016b) created a system of multiple GAN pairs that interchanged during training. These papers showed that GANs with additional models is a promising evolution and should be further explored. Hence we propose Generative Multi-Adversarial Network with Historic Discriminator (GMAN-HD), a GAN architecture with a second discriminative model. The additional discriminative network is trained on a record consisting of previously generated samples. GMAN-HD is inspired by the success of using additional Discriminator networks (Durugkar et al., 2016; Im et al., 2016b). The historical archive is inspired by the archive used by the Discriminator in Shrivastava et al. (2016).

The historic archive is the same size as a regular batch. At each iteration, a minimum of a quarter and up to one half of the archive is randomly replaced with the best images selected through ABNS with audition size equal a regular batch. The archive was originally much larger with images
spanning the last epochs. This did not perform well and the archive was changed to be more up to
date. The audition size was kept low to increase performance. The Generator is trained against both
Discriminators each iteration. The regular Discriminator is trained in usual fashion, but the His-
toric Discriminator is trained against the training data and the images in the archive. The Historic
Discriminator is never shown the current output of the Generator.

4.3 Evaluation metrics

[Theis et al. (2016)] concluded that both the training and evaluation of a GAN model should be
modelled after the target application which for this thesis is image generation. Using log-likelihoods
as an evaluation metric for GANs was first introduced by Goodfellow [Goodfellow et al. (2014)]).
However, as stated in both [Goodfellow et al. (2014)] and [Theis et al. (2016)], log-likelihood is not
suitable for evaluation of higher dimensional data, such as images, and has largely been phased
out as GANs have evolved. Similarly, using nearest neighbour algorithm to compare pixel data in
images was suggested by [Durugkar et al. (2016)] and [Ledig et al. (2016)], but [Theis et al. (2016)]
argued this was not sufficient. [Salimans et al. (2016)] used a human annotator to evaluate the quality
of their GAN. However, conducting a visual Turing test would is expensive and time consuming.

Generative Adversarial Metric (GAM) [Im et al., 2016a] is a general method for comparing two
GANs. GAM does not directly evaluate the quality of image generation, but compares two trained
GAN models by evaluating each other. It is however not perfect since it does not evaluate im-
age quality compared to the training dataset. As [Salimans et al. (2016)] noted when comparing
Feature matching and Minibatch discrimination, certain techniques may create a better class-
ifier(Discriminators) without improving image quality. Still, GAM is the best suited evaluation
metric for this work and a general framework means that GAM can be customized to the desired
task.

The custom implementation of GAM is named Serial Generative Adversarial Metric (SGAM). Serial
refers to the fact that checkpoints of the GAN models are never loaded at the same time, but one
at time in series. The main reason behind this is memory consumption. Keeping two models in
memory at the same time requires double the GPU memory. The generated samples are stored in the
systems main memory. All experiments and evaluations are run on the same computer with relatively
low amount of GPU memory for such task. Running the two models in series is therefore a simple
trick to guarantee there is always enough available GPU memory. TensorFlow has built in support for
saving and restoring checkpoints of a previously trained model. Loading a model is generally a quite
efficient and fast process. However, implementing GAM through the equations in (Im et al., 2016a)
resulted in an unstable evaluation. The min max battle of GANs separates them from conventional
ANNs. The models are constantly evolving caused by the min-max battle. The solution to this
problem was simply to modify the equations to use classification accuracy and not the error. This
helps stabilize the metric when error rates are low, but still fails if the accuracies are too different.

5 Results

Serial Generative adversarial metric (SGAM) is the main evaluation metric. Each technique will be
compared to against a default implementation of DCGAN. The final output score will not the sole
determining factor. SGAM calculates local result after each epoch and the results of these will also
be a point of discussion. A technique may for instance be better during the early epochs, but worse
overall. Some manual inspection of the generated samples will also be conducted to highlight some
of the variations between techniques. To maintain a balance between performance and accuracy, the
test accuracy and sample accuracy are calculated by classifying 12,88 batches, equal to 200 batches.

5.1 Static reusable noise

The SGAM results for comparing models trained with SRN against regular DCGAN are shown in
Table 2. The effect of SRN varied depending on the dataset, but did not have a large effect on
performance. Note that the model’s performance was not affected by whether random noise or the
noise used during training was used to generate samples. When trained on the CelebA dataset, the
model using SRN won both globally and was most local victorious. CIFAR-10 trained with SRN
performed worse overall, but won more local victories. Models trained using the CIFAR-10 dataset
Table 2: Comparison of regular DCGAN and DCGAN trained with Static Reusable Noise

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Epochs</th>
<th>GAN 1</th>
<th>GAN 2</th>
<th>$r_{test}$</th>
<th>$r_{sample}$</th>
<th>Winner</th>
</tr>
</thead>
<tbody>
<tr>
<td>CelebA</td>
<td>16</td>
<td>DCGAN</td>
<td>SRN$_{Z=\text{random}}$</td>
<td>0.9867</td>
<td>0.9981</td>
<td>SRN$_{Z=\text{random}}$</td>
</tr>
<tr>
<td></td>
<td>16</td>
<td>DCGAN</td>
<td>SRN$_{Z=\text{training}}$</td>
<td>0.9875</td>
<td>0.9984</td>
<td>SRN$_{Z=\text{training}}$</td>
</tr>
<tr>
<td>IKEA</td>
<td>22</td>
<td>DCGAN</td>
<td>SRN$_{Z=\text{random}}$</td>
<td>0.9981</td>
<td>4.5108</td>
<td>DCGAN</td>
</tr>
<tr>
<td></td>
<td>22</td>
<td>DCGAN</td>
<td>SRN$_{Z=\text{training}}$</td>
<td>0.9980</td>
<td>4.5106</td>
<td>DCGAN</td>
</tr>
<tr>
<td>CIFAR-10</td>
<td>50</td>
<td>DCGAN</td>
<td>SRN$_{Z=\text{random}}$</td>
<td>1.0026</td>
<td>1.0571</td>
<td>DCGAN</td>
</tr>
<tr>
<td></td>
<td>50</td>
<td>DCGAN</td>
<td>SRN$_{Z=\text{training}}$</td>
<td>1.0019</td>
<td>1.0556</td>
<td>DCGAN</td>
</tr>
</tbody>
</table>

Table 3: Regular DCGAN and DCGAN trained with Image Based Noise Generation (IBNG)

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Epochs</th>
<th>GAN 1</th>
<th>GAN 2</th>
<th>$r_{test}$</th>
<th>$r_{sample}$</th>
<th>Winner</th>
</tr>
</thead>
<tbody>
<tr>
<td>CelebA</td>
<td>18</td>
<td>DCGAN</td>
<td>IBNG$_{Z=\text{random}}$</td>
<td>0.9762</td>
<td>1.0101</td>
<td>DCGAN</td>
</tr>
<tr>
<td></td>
<td>18</td>
<td>DCGAN</td>
<td>IBNG$_{Z=\text{training}}$</td>
<td>0.9742</td>
<td>1.010</td>
<td>DCGAN</td>
</tr>
<tr>
<td>IKEA</td>
<td>20</td>
<td>DCGAN</td>
<td>IBNG$_{Z=\text{random}}$</td>
<td>0.9656</td>
<td>1.2667</td>
<td>DCGAN</td>
</tr>
<tr>
<td></td>
<td>20</td>
<td>DCGAN</td>
<td>IBNG$_{Z=\text{training}}$</td>
<td>0.9597</td>
<td>1.2203</td>
<td>DCGAN</td>
</tr>
<tr>
<td>CIFAR-10</td>
<td>25</td>
<td>DCGAN</td>
<td>IBNG$_{Z=\text{training}}$</td>
<td>0.9437</td>
<td>1.0473</td>
<td>DCGAN</td>
</tr>
<tr>
<td></td>
<td>50</td>
<td>DCGAN</td>
<td>IBNG$_{Z=\text{random}}$</td>
<td>0.9888</td>
<td>1.0239</td>
<td>DCGAN</td>
</tr>
<tr>
<td></td>
<td>50</td>
<td>DCGAN</td>
<td>IBNG$_{Z=\text{training}}$</td>
<td>0.9872</td>
<td>1.0083</td>
<td>DCGAN</td>
</tr>
</tbody>
</table>

usually vary immensely and any technique should make more of an impact on the SGAM ratios. SRN did not perform well when trained on the IKEA dataset. The Discriminator appears to be too powerful and is quickly unable to correctly classify the samples generated from regular DCGAN.

5.2 IMAGE BASED NOISE GENERATION

The results of comparing models trained with IBNG with regular DCGAN are shown in Table 2. The results show that IBNG created a worse model that was more overfitted to the training data. Each model was compared to both random noise and noise generated with IBNG. Of the two, the familiar noise of IBNG performed slightly better, but still worse than regular DCGAN. Using the CelebA dataset with IBNG resulted in a worse model. CIFAR-10 has widely different images and as a result the noise samples are mostly different as well. Using IBNG is essentially the same as SRN.

The most interesting result was with models trained on the IKEA dataset, Figure 2. As with SRN, the sample score does eventually collapse. Images in IKEA dataset have much common with each other. Using IBNG, this aspect is helpful during the early epochs of training, but a curse later. During the first six epochs, the samples generated from the model trained using IBNG is superior and able to correctly classify every sample from the regular DCGAN. The performance of both models begins to align after about four epochs of training. Regular DCGAN is able to stay at this performance level, but after ten epochs, IBNG begins to fail at classifying the images from regular DCGAN. The model begins to collapse as the Discriminator become too strong for the Generator. At the same time, regular DCGAN has constantly improved and performing. The peak of both sample scores at epoch 19 is interesting. The likely cause is that the generator trained IBNG are generating images of so poor quality regular DCGAN fails to accurately classify them.

5.3 AUDITION BASED NOISE SELECTION

The results of the models trained using Audition Based Noise Selection (ABNS) are shown in Table 2. The usual problems observed with IKEA dataset occurred with ABNS as well. Otherwise, ABNS improved performance somewhat for CelebA and considerably for CIFAR-10. ABNS made the already computationally expensive training process even worse. This limited the number of experiments conducted for CelebA and IKEA datasets. CIFAR-10 requires considerably less time and using this dataset was therefore prioritized.
Under review as a conference paper at ICLR 2018

Figure 2: Performance of regular DCGAN and DCGAN with IBNG trained on the IKEA dataset

Table 4: Regular DCGAN and DCGAN trained with Audition Based Noise Selection (ABNS)

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Epochs</th>
<th>GAN 1</th>
<th>GAN 2</th>
<th>$r_{\text{test}}$</th>
<th>$r_{\text{sample}}$</th>
<th>Winner</th>
</tr>
</thead>
<tbody>
<tr>
<td>CelebA</td>
<td>18</td>
<td>DCGAN</td>
<td>ABNS-best$_3$</td>
<td>1.0815</td>
<td>0.9989</td>
<td>ABNS-best$_3$</td>
</tr>
<tr>
<td></td>
<td>15</td>
<td>DCGAN</td>
<td>ABNS-mixed$_3$</td>
<td>1.0690</td>
<td>0.9987</td>
<td>ABNS-mixed$_3$</td>
</tr>
<tr>
<td>IKEA</td>
<td>8</td>
<td>DCGAN</td>
<td>ABNS-mixed$_3$</td>
<td>0.8715</td>
<td>1.7889</td>
<td>DCGAN</td>
</tr>
<tr>
<td></td>
<td>14</td>
<td>DCGAN</td>
<td>ABNS-mixed$_6$</td>
<td>0.9630</td>
<td>2.0341</td>
<td>DCGAN</td>
</tr>
<tr>
<td>CIFAR-10</td>
<td>40</td>
<td>DCGAN</td>
<td>ABNS-best$_3$</td>
<td>1.0053</td>
<td>0.9129</td>
<td>ABNS-best$_3$</td>
</tr>
<tr>
<td></td>
<td>40</td>
<td>DCGAN</td>
<td>ABNS-best$_6$</td>
<td>1.0517</td>
<td>0.8508</td>
<td>ABNS-best$_6$</td>
</tr>
<tr>
<td></td>
<td>50</td>
<td>DCGAN</td>
<td>ABNS-mixed$_3$</td>
<td>1.0054</td>
<td>0.7805</td>
<td>ABNS-mixed$_3$</td>
</tr>
<tr>
<td></td>
<td>40</td>
<td>DCGAN</td>
<td>ABNS-mixed$_6$</td>
<td>0.9683</td>
<td>0.743</td>
<td>ABNS-mixed$_6$</td>
</tr>
<tr>
<td></td>
<td>40</td>
<td>ABNS-mixed$_6$</td>
<td>ABNS-mixed$_3$</td>
<td>0.9215</td>
<td>0.9059</td>
<td>ABNS-mixed$_3$</td>
</tr>
<tr>
<td></td>
<td>40</td>
<td>ABNS-best$_6$</td>
<td>ABNS-mixed$_6$</td>
<td>0.9215</td>
<td>0.9059</td>
<td>ABNS-mixed$_6$</td>
</tr>
</tbody>
</table>

ABNS was able to improve performance slightly when trained on the CelebA dataset. Both mixed and best mode won the global victory, but the local victories were more divided. Figure 3 plots the results of the SGAM comparison between ABNS in best mode and Figure 4 shows the result of mixed mode. Both were trained with audition size of 3 times the batch size of 64. ABNS-mixed$_3$ is more stable overall, but ABNS-best$_3$ is noticeably better during the first six epochs. ABNS-mixed$_3$ is still performing better than DCGAN during the early epochs. The results after that point are more varied with regular DCGAN often being the best performing model.

Training on CIFAR-10 was greatly improved with ABNS. The table shows that increasing audition size increases performance. All variations of ABNS severely outperform regular DCGAN. The results from SGAM when comparing the best performing model, ABNS-mixed$_6$ against regular DCGAN are plotted in Figure 5. Comparing sample scores reveals ABNS-mixed$_6$ to severely outperform DCGAN up until epoch 30. At this point DCGAN is slightly outperforming ABNS-mixed$_6$.

Using ABNS with the IKEA dataset once again caused the Discriminator to become overpowered. Neither of the two modes nor increasing the auditions size was able to prevent the collapse. Figure 6 plots the results of ABNS in mixed mode with audition size of 3. The collapse is clearly visible at epoch 4. The Generator makes some strong attempts at recuperating between epoch 6 and 10, but the strong Discriminator eventually regains total control. Inspecting both the off-balance test ratio in Figure 4 and the plotted test score in Figure 6 indicates the ABNS-mixed$_6$ made the Discriminator more overfitted towards the training data.

5.4 Generative Multi-Adversarial Network with Historic Discriminator

Generative Multi-Adversarial Network with Historic Discriminator (GMAN-HD) performed well against regular DCGAN when trained on CelebA and CIFAR-10. GMAN-HD was also tested against models trained using ABNG. The results are shown in Table 5. Note that only the main Discriminator was used for SGAM comparisons.
Figure 3: Comparison between DCGAN and ABNS-best trained on the CelebA dataset

Figure 4: Comparison between DCGAN and ABNS-mixed trained on the CelebA dataset

Figure 5: Comparison between DCGAN and ABNS-mixed trained on CIFAR-10 datasets

Figure 6: Performance of regular DCGAN versus ABNS-mixed trained on the IKEA dataset
Table 5: SGAM comparison between regular DCGAN and DCGAN trained with Multi-Adversarial Network with Historic Discriminator

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Epochs</th>
<th>GAN 1</th>
<th>GAN 2</th>
<th>$r_{test}$</th>
<th>$r_{sample}$</th>
<th>Winner</th>
</tr>
</thead>
<tbody>
<tr>
<td>CelebA</td>
<td>14</td>
<td>DCGAN</td>
<td>GMAN-HD</td>
<td>1.0728</td>
<td>0.9980</td>
<td>GMAN-HD</td>
</tr>
<tr>
<td></td>
<td>14</td>
<td>ABNS-mixed$_3$</td>
<td>GMAN-HD</td>
<td>0.9983</td>
<td>0.9989</td>
<td>GMAN-HD</td>
</tr>
<tr>
<td>IKEA</td>
<td>19</td>
<td>DCGAN</td>
<td>GMAN-HD</td>
<td>0.9734</td>
<td>2.9231</td>
<td>DCGAN</td>
</tr>
<tr>
<td></td>
<td>8</td>
<td>ABNS-mixed$_3$</td>
<td>GMAN-HD</td>
<td>1.207</td>
<td>1.7083</td>
<td>ABNS-mixed$_3$</td>
</tr>
<tr>
<td></td>
<td>14</td>
<td>ABNS-mixed$_6$</td>
<td>GMAN-HD</td>
<td>1.0289</td>
<td>1.5507</td>
<td>ABNS-mixed$_6$</td>
</tr>
<tr>
<td>CIFAR-10</td>
<td>46</td>
<td>DCGAN</td>
<td>GMAN-HD</td>
<td>1.0058</td>
<td>0.8194</td>
<td>GMAN-HD</td>
</tr>
<tr>
<td></td>
<td>40</td>
<td>GMAN-HD</td>
<td>ABNS-best$_3$</td>
<td>0.9964</td>
<td>1.2095</td>
<td>Historic</td>
</tr>
<tr>
<td></td>
<td>46</td>
<td>GMAN-HD</td>
<td>ABNS-mixed$_3$</td>
<td>0.9984</td>
<td>0.9222</td>
<td>ABNS-mixed$_3$</td>
</tr>
<tr>
<td></td>
<td>40</td>
<td>GMAN-HD</td>
<td>ABNS-mixed$_6$</td>
<td>0.9610</td>
<td>0.9576</td>
<td>ABNS-mixed$_6$</td>
</tr>
</tbody>
</table>

Figure 7: Comparison of GMAN-HD vs. DCGAN on the CelebA dataset

The Historic Discriminator did not perform well on the IKEA dataset. The Discriminator collapses after just two epochs of training. Still, the Generator and second Discriminator do make some attempts at increasing performance. The sample accuracy begins to rise at the 11th epoch. The tests scores are similar. The DCGAN test score has a peak at epoch 16, while GMAN-HD’s accuracy stays close to 100% from epoch 10. The results of training Historic Discriminator on the CelebA dataset resembles a mix between ABNG in best mode and mixed mode. The sample score is constantly above the score of DCGAN until a small peak at epoch 13. The results from CIFAR-10 were interesting. The samples scores are almost identical until epoch 10 when HD’s score suddenly begins to rise. The score fluctuates for some time before stabilizing at around 99.9% accuracy after 18 epochs. DCGAN is not able to catch up until 10 epochs later and even then, HD’s sample accuracy is still higher. The test scores are similar throughout, but HD follows the same patterns as with ABNS, creating a weaker but more stable Discriminator.

Figure 8: Comparison of GMAN-HD vs. DCGAN on the CIFAR-10 dataset
6 DISCUSSION

It is clear from the experiments that the dataset used for training had a large impact on the results. This result will therefore first be discussed overall and then in relation to each dataset.

SRN is a simple technique which reuses the same noise every epoch. SRN was able to increase accuracy slightly on CelebA and CIFAR-10 during the earlier epochs when most the learning occurs. SRN in its current form is not a recommended technique, but highlights the potential for reducing the range of the Generator input to stabilize the learning process.

IBNG is similar to SRN and generates the input noise from images in the training set. The goal behind the technique is to make the learning task easier for the Generator, but IBNG had the opposite effect and created a too strong Discriminator that was more overfitted towards the training data and performed slightly worse than regular DCGAN. Although IBNG still caused the model to collapse when trained on the IKEA dataset, the collapse occurred much later in the training process than any of the other techniques.

ABNS is a more advanced and computationally expensive technique that tasks the Discriminator to pick the most representable images to guide training. Using ABNS when trained on CelebA performed similar to SRN. Experimenting with larger audition sizes may improve performance further. This is interesting as SRN is a much more simple technique. ABNS was more successful on CIFAR-10, a more difficult dataset. All variations of ABNS were able to drastically increase performance. Testing on CIFAR-10 saw an increase in performance with an increase in audition size.

The final proposed technique added an additional Discriminator to the DCGAN architecture. This GMAN-HD Discriminator was trained on past, well rated images from the Generator. While requiring less training time, GMAN-HD performed better than the other techniques on the CelebA dataset, winning both globally and almost all local victories. HD did also perform well on CIFAR-10 dataset, but was beat by some variations of ABNS.

While SRN and IBNG did perform well on under certain conditions, overall, the techniques are too simple to warrant further use. Both of these technique were proposed to help the Generator during training by putting limitations on its input. This did to a lesser degree stabilized training during the early epochs, but usually ended up limiting the model overall. ABNS had better success at creating a stronger Generator as it does not limit the available range of the input. ABNS showed that the learning process could be improved by maximize learning each iteration. GMAN-HD achieved similar results on CIFAR-10 and CelebA. The models trained with GMAN-HD was not as powerful as ABNS with larger auditions sizes, but proved to be more stable against unaltered DCGAN.

6.1 CELEBA DATASET

The CelebA dataset rated as an easy dataset for use with GAN, subsection refer sec:publicdatasets. The dataset has a large number of images with common features of a face centrally aligned in frame. Regular DCGAN is more than able to learn these features through unsupervised learning. Figure 10 plots the classification accuracy for the Discriminator
model in regular DCGAN when training on CelebA dataset. The accuracy converges after about four epochs of training. Being an easy dataset made the proposed techniques have less of impact on performance. Although small, three of techniques were able to consistently improve performance on CelebA. The largest improvements were seen during the early of epochs of training.

6.2 CIFAR-10 DATASET

CIFAR-10 was the most convenient dataset used for the experiments. The dataset has fewer images and lower resolution which requires less training time than the other two datasets. CIFAR-10 was rated as a hard dataset in the previous chapter and this proved accurate. Figure 11 shows how DCGAN struggles during the early training phase and requires several dozen epochs of training before be able to accurately classifying generated and real images correctly. With CIFAR-10 being a harder dataset, some of the suggested techniques were better able to increase performance when using it as training data. The simpler techniques of SRN and IBNG did limit the performance somewhat. The more advanced techniques of ABNG and MGAN with historic Discriminator were more successful.

6.3 IKEA DATASET

The IKEA dataset was created during the pre-study. The dataset was rated at medium difficult in the previous chapter, but using it for this experiments seems to indicate it is a harder dataset than first anticipated. Regular DCGAN is able to capture the white background and overall create quite sharp images. The images do however never reach a point where they appear natural. This may be caused the images having few consistent features among them except for the white background. The dataset is also being artificially augmented to increase its size. For this dataset, each original image was copied and slightly adjust 100 times which may impact learning in some way. Figure 13 plots the classification accuracy of the Discriminator when regular DCGAN is trained on the dataset. Notice that the DCGAN requires about five to seven epochs of training before reaching 100% accuracy.

None of the proposed techniques were able to consistently improve the models trained on the IKEA dataset. Every attempt ended with the models sample score eventually collapsing. The collapse oc-
curs after about five to seven epochs which correlates with Discriminator performance of Figure 13. This collapse was surprising and conflicting with the results of the other datasets. Interestingly, it was only the sample score of the afflicted model that collapse, the test score was usually high. The sample score of the other model, evaluating the generated images from the afflicted model, was usually above 90%.

Figure 12 shows samples generated with ABNS and shows samples generated before and after the collapse. The number next to each sample represent the rating of the models Discriminator. Rating above 50 is believed to be real images and rating under is believed to be generated. At epoch 4 the performance of the Discriminator and generates is well balanced. The Generator is able generated decent samples that often fool the Discriminator. These images provide valuable feedback on how to improve further. At epoch six, the Discriminator has become too strong as seen by all rating being below 50 and most even being below 30.

It should be noted that this sort of collapse does occur on occasion with other datasets, but that is only between a single iteration. A strong Discriminator halts the learning of the Generator as none of its improvements is able to fool the Generator. It is reasonable to believe a strong Discriminator should result in high sample score. In fact, the opposite is true. The Discriminator is only taught how generated samples should look from its own Generator. A strong Discriminator leads to a poor Generator which produces low quality samples. The Discriminator is taught that these low quality samples represents all generated samples. This is of course not the case when shown samples from another model with better balanced models. These samples will have much higher quality than those of its own Generator and thus believe they to be real. This causes the collapse.

The analysis of the results seems to indicate an unbalanced and overpowered Discriminator to be the root of the problem. Training on the IKEA dataset is a delicate process. The images in the IKEA dataset appears to be easier to learn for the Discriminator then the Generator. All the suggested techniques ended up make the task easier for the Discriminator. After about five epochs of training, the Discriminator has become so skilled at its task of separating training data from generated samples the Generator is unable to progress.

7 Conclusion and Future Work

Three main techniques were proposed to investigate how performance of a Generative Adversarial Network is affected by altering the input to be more favourable towards the Generator. Each technique makes guided modifications to the otherwise random input of the generative model to make the learning task easier. Reusing the input of the Generators through Static Reusable Noise was a simple technique which did affect the performance during training in a positive, albeit minor, way. The noise was only repeated once every epoch of training which may be too infrequent. Still, tests using Image Based Noise Generation showed that making the input too similar only limits the
model. The most promising of the three suggested techniques was Audition Based Noise Selection (ABNS). This technique allows the Discriminator to select representable noise samples that help guide training. Experiments conducted with difficult datasets showed that ABNS was able to guide the training through the most difficult parts of learning. ABNS had less effects on simpler datasets. Comparison between regular models showed that ABNS makes for a Discriminator that is overall weaker but more stable. The three techniques show that guided alterations to the input of the Generator has the potential of increasing performance throughout the learning process, but not overall. Regularly trained GAN models were eventually able to reach similar performance on all datasets.

Expanding GAN with additional ANNs is a popular area of research. The common approach so far has been to add identical additional networks with slightly varying parameters. This paper has examined the effect of GAN models were the models have varying training data. The suggested Generative Multi-Adversarial Network with Historic Discriminator (GMAN-HD) uses two Discriminators that are trained separately and on different data, thereby asymmetric. The Historic Discriminator is trained on previous well-rated output from the Generator. GMAN-HD creates a model that outperforms regular GAN models on multiple datasets. The Discriminators are more stable and less overfitted towards the training data. The largest gap in performance was seen early when most of learning occurs.

The experiments showed how much the training set affected the results. ABNS and GMAN-HD saw the largest increase in performance on the most difficult dataset tested, CIFAR-10. Future work should verify the results on other datasets as well. ABNS and GMAN-HD helps guide the learning process which may enable even more difficult datasets to be used with GANs. Further, using two asymmetrically trained Discriminators was shown to increase performance and create a more stable model. GMAN-HD was just one concrete implementation of this method. Additional experimentation could be conducted with an increasing number of models.

![Figure 13: Classification accuracy for DCGAN on the IKEA dataset](image)
REFERENCES


16