SMALL-GAN: SPEEDING UP GAN TRAINING USING CORE-SETS

Anonymous authors
Paper under double-blind review

ABSTRACT

Recent work by Brock et al. (2018) suggests that Generative Adversarial Networks (GANs) benefit disproportionately from large minibatch sizes. The batch sizes in (Brock et al., 2018) are slow and expensive to emulate on conventional hardware. Thus, it would be nice if there were some method by which we could generate batches that were *effectively big* though small in practice. In this work, we propose such a method, inspired by the use of Coreset-selection in active learning. When training a GAN, we draw a large batch of samples from the prior and then compress that batch using Coreset-selection. To create effectively large batches of real images, we create a cached dataset of Inception activations of each training image, randomly project them down to a smaller dimension, and then use Coreset-selection on those projected embeddings at training time. We conduct experiments showing that this technique substantially reduces training time and memory usage for modern GAN variants, that it reduces the fraction of dropped modes in a synthetic dataset, and that it helps us use GANs to reach a new state of the art in anomaly detection.

1 INTRODUCTION

Generative Adversarial Networks (GANs) (Goodfellow et al., 2014) have become a popular research topic. Arguably the most impressive results have been in image synthesis (Brock et al., 2018; Salimans et al., 2018; Miyato et al., 2018; Zhang et al., 2018; 2017), but they have also been applied fruitfully to text generation (Fedus et al., 2018; Guo et al., 2018), domain transfer learning (Zhu et al., 2017; Zhang et al., 2017; Isola et al., 2017), and various other tasks (Xian et al., 2018; Ledig et al., 2017; Zhu & Bento, 2017).

Recently, Brock et al. (2018) substantially improved the results of Zhang et al. (2018) by using very large mini-batches during training. The effect of large mini-batches in the context of deep learning is well-studied (Smith et al., 2017; Goyal et al., 2017; Keskar et al., 2016; Shallue et al., 2018) and general consensus is that they can be helpful in many circumstances, but the results of Brock et al. (2018) suggest that GANs benefit disproportionately from large batches. In fact, Table 1 of Brock et al. (2018) shows that for the Frechet Inception Distance (FID) metric (Heusel et al., 2017) on the ImageNet dataset, the scores can be improved from 18.65 to 12.39 simply by making the batch eight times larger.

Unfortunately, increasing the batch size in this manner is not always possible since it increases the computational resources required to train these models - beyond the reach of conventional hardware. The experiments from the BigGAN paper require a full 'TPU Pod'. The 'unofficial' open source release of BigGAN achieves this by accumulating gradients across 8 different V100 GPUs and only taking an optimizer step every 8 gradient accumulation steps. Future research on GANs would be much easier if we could have the gains from large batches without these pain points. In this paper, we take steps toward accomplishing that goal by proposing a technique that allows for *mimicking* large batches without the computational costs of actually using larger batch-sizes.

We aim to maximally retain the information from a large batch when using a smaller batch. One effective way to perform subsampling to form the smaller batch is by using Core-set selection (Agarwal et al., 2005). Our method samples large candidate batches and then selects smaller Core-sets from them to be used in training. Intuitively, we want the small batches to have similar 'coverage' to the large batch, especially coverage of the many modes of the data distribution. This technique yields many of the benefits of having large batches with much less computational overhead. Moreover, it is generic, and so can be applied to nearly all GAN variants. We plan to release the code and cached datasets to ensure reproducibility of our work.

Our contributions can be summarized as follows:

- We introduce a simple, computationally cheap method to increase the 'effective batch size' of GANs, which can be applied to any GAN variant.
- We conduct experiments on the CIFAR and LSUN datasets showing that our method can substantially improve on the FID scores across different GAN architectures given a fixed batch size.
- We use our method to improve the performance of the technique from Kumar et al. (2019) on the task of anomaly detection.

2 BACKGROUND AND NOTATION

Generative Adversarial Networks A Generative Adversarial Network (or GAN) is a system of two neural networks trained in an adversarial manner. The generator, G, takes as input samples from a prior $z \sim p(z)$ and outputs the learned distribution, $G(z)$. The discriminator, D, receives as input both the training examples, X, and the synthesized samples, $G(z)$, and outputs a distribution $D(.)$ over the possible sample source. The discriminator is then trained to maximize the following objective:

$$\mathcal{L}_D = -\mathbb{E}_{x \sim p_{\text{data}}}[\log D(x)] - \mathbb{E}_{z \sim p(z)}[\log(1 - D(G(z)))] \tag{1}$$

while the generator is trained to minimize[1]:

$$\mathcal{L}_G = -\mathbb{E}_{z \sim p(z)}[\log D(G(z))] \tag{2}$$

Put simply, the generator is trained to *trick* the discriminator into believing that the generated samples $G(z)$ actually come from the target distribution, $p(x)$, while the discriminator is trained to be able to distinguish the samples from each other.

Inception Score and Frechet Inception Distance: We will refer frequently to the Frechet Inception Distance (FID) (Heusel et al., 2017), to measure the effectiveness of an image synthesis model. To compute this distance, one assumes that we have a pre-trained Inception classifier. One further assumes that the activations in the penultimate layer of this classifier come from a multivariate Gaussian. If the activations on the real data are $N(m, C)$ and the activations on the fake data are $N(m_w, C_w)$, the FID is defined as:

$$\|m - m_w\|_2^2 + \mathrm{Tr}\big(C + C_w - 2\big(CC_w\big)^{1/2}\big) \tag{3}$$

Core-set selection: In computational geometry, a Core-set, Q, of a set P is a subset $Q \subset P$ that approximates the 'shape' of P (Agarwal et al., 2005). Core-sets are used to quickly generate approximate solutions to problems whose full solution on the original set would be burdensome to compute. Given such a problem[2] , one computes Q, then quickly computes the solution to the problem for Q and converts that into an approximate solution for the original set P. The general Core-set selection problem can be formulated several ways, but here we use the the *minimax facility location* formulation (Farahani & Hekmatfar, 2009):

$$\min_{Q:|Q|=k} \max_{x_i \in P} \min_{x_j \in Q} d(x_i, x_j) \tag{4}$$

where k is the desired size of Q, and $d(.,.)$ is a metric on P. Intuitively, the formula above encodes the following objective: Find some set, Q, of points of size k such that the maximum distance

---

[1] This is the commonly used "Non-Saturating Cost". There are many others, but for brevity and since our technique we describe is agnostic to the loss function, we will omit them.

[2] As an example, consider computing the diameter of a point-set (Agarwal et al., 2005)

between a point in P and its nearest point in Q is minimized. Since finding the exact solution to the minimax facility location problem is NP-Hard (Wolsey & Nemhauser, 2014), we will have to make do with a greedy approximation, detailed in Section 3.3.

Algorithm 1 `GreedyCoreset`

```
Input: batch size ($k$), data points ($x$ where $|x| > k$)
Output: subset of $x$ of size $k$
  $s \leftarrow \{\}$                                          ▷ Initialize the sampled set
  while $|s| < k$ do                                     ▷ Iteratively add points to sampled set
      $p \leftarrow \arg\max_{x_i \notin s} \min_{x_j \in s} d(x_i, x_j)$
      $s \leftarrow s \cup \{p\}$
  end whilereturn $s$
```

3 USING CORE-SET SAMPLING FOR GANS (OR SMALL-GAN)

We aim to use Core-set sampling to increase the effective batch size during GAN training. This involves replacing the basic sampling operation that is done implicitly when minibatches are created. This implicit sampling operation happens in two places: First, when we create a minibatch of samples drawn from the prior distribution $p(z)$. Second, when we create a minibatch of samples from the target distribution $p_{\texttt{data}}(x)$ to update the parameters of the discriminator. The first of these replacements is relatively simple, while the second presents challenges. In both cases, we have to work around the fact that actually doing Core-set sampling is computationally hard.

3.1 SAMPLING FROM THE PRIOR

We need to sample from the prior when we update the discriminator and generator parameters. Our Core-set sampling algorithm doesn't take into account the geometry of the space we sample from, so sampling from a complicated density might cause trouble. This problem is not intractable, but it's nicer not to have to deal with it, so in the absence of any evidence that the form of the prior matters very much, we define the prior in our experiments to be the uniform distribution over a hypercube. To add Core-set sampling to the prior distribution, we sample n points from the prior, where n is greater than the desired batch size, k. We then perform Core-set selection on the large batch of size n to create a batch of size k. The smaller batch is what's actually used to perform an SGD step.

3.2 SAMPLING FROM THE TARGET DISTRIBUTION

Sampling from the target distribution is more challenging. The elements drawn from the distribution are high dimensional images, so taking pairwise distances between them will tend to work poorly due to concentration of distances (Donoho et al., 2000; Sinha et al., 2019), and the fact that euclidean distances are semantically meaningless in image space (Girod, 1993; Eskicioglu & Fisher, 1995).

To avoid these issues, we instead pre-process our data set by computing the 'Inception Embeddings' of each image using a pre-trained classifier (Szegedy et al., 2017). This is commonly done in the transfer-learning literature, where it is generally accepted that these embeddings have nontrivial semantics (Yosinski et al., 2014). Since this pre-processing happens only once at the beginning of training, it doesn't affect the per-training-step performance. In practice this step can be performed and the resultant embeddings can be saved since $p_{\texttt{data}}(x)$ remains constant.

In order to further reduce the time taken by the Core-set selection procedure, and inspired by the Johnson-Lindenstrauss Lemma (Dasgupta & Gupta, 2003), we can take random low dimensional projections of the Inception Embeddings, while preserving distances. These two techniques together give us low dimensional representations of the training set images in which pairwise Euclidean distances have meaningful semantics. We can then use Core-set sampling on those representations to select images at training time, analogous to how we select images from the prior.

3.3 GREEDY CORE-SET SELECTION

In the above sections, we have invoked Core-set selection while glossing over the detail that exactly solving the k-center problem is NP-hard. This is important, because we propose to use Core-set selection at *every* training step[3]. Fortunately, we can make do with an approximate solution which is more computationally efficient using a greedy algorithm. Toward that end, we use the greedy k-center algorithm similar to Sener & Savarese (2017) which is summarized in Alg. 1.

3.4 PUTTING IT ALL TOGETHER

Our full proposed algorithm for GAN training is presented in Alg. 2. More implementation details and design choices are presented in Section 4. Our technique is agnostic to the underlying GAN framework and therefore can replace random sampling of mini-batches for all GAN variants. We choose to experiment with a few different recently proposed GAN models. Our method can also be employed alongside any other optimization scheme used to further reduce the variance in GANs.

Algorithm 2 `Small-GAN`

```
Input: target batch size ($k$), starting batch size ($n > k$), Inception embeddings ($\phi_I$)
Output: a trained GAN
  Initialize networks $G$ and $D$
  for $step = 1$ to ... do
    $z \sim p(z)$                                              ▷ Sample $n$ points from the prior
    $x \sim p(x)$                                              ▷ Sample $n$ points from the data distribution
    $\phi(x) \leftarrow \phi_I(x)$                             ▷ Get cached embeddings for $x$
    $\hat{z} \leftarrow$ GreedyCoreset($z$)                    ▷ Get Core-set of $z$
    $\widehat{\phi(x)} \leftarrow$ GreedyCoreset($\phi(x)$)    ▷ Get Core-set of embeddings
    $\hat{x} \leftarrow \phi_I^{-1}(\widehat{\phi(x)})$        ▷ Get $x$ corresponding to sampled embeddings
    Update GAN parameters as usual
  end for
```

4 EXPERIMENTS

In this section we look at the performance of our proposed sampling method on various tasks: In the first experiment, we train a GAN on a Gaussian mixture dataset with a large number of modes and confirm our method substantially mitigates 'mode-dropping'. In the second, we apply our technique to GAN-based anomaly detection (Kumar et al., 2019) and significantly improve on prior results. Finally, we test our method on standard image synthesis benchmarks and confirm that our technique seriously reduces the need for large mini-batches in GAN training. The variety of settings in these experiments testifies to the generality of our proposed technique.

4.1 IMPLEMENTATION DETAILS

For our Core-set algorithm, the distance function, $d(\cdot, \cdot)$ s the ℓ_2-norm for both the prior and target distributions. The hyper-parameters used in each experiment are the same as originally proposed in the paper introducing that experiment, unless stated otherwise. For over-sampling, we use a factor of 4 for the prior $p(z)$ and a factor of 8 for the target, $p(x)$, unless otherwise stated. We investigate the effects of different over-sampling factors in the ablation study in Section 4.6.

4.2 MIXTURE OF GAUSSIANS

We first investigate the problem of mode dropping (Arora et al., 2018) in GANs, where the GAN generator is unable to recover some modes from the target data set. We investigate the performance of training a GAN to recover a different number of modes of 2D isotropic Gaussian distributions,

---

[3] Though the Core-set sampling does happens on CPU and so could be done in parallel to the GPU operations used to train the model, as long as the Core-set sampling time doesn't exceed the time of a forward and backward pass – which it doesn't.

Number of Modes	% of Recovered Modes (GAN)	% of Recovered Modes (Ours)	% of High-Quality Samples (GAN)	% of High-Quality Samples (Ours)
25	**100**	**100**	95.76	**98.9**
36	**100**	**100**	92.73	**95.34**
49	98.12	**99.85**	84.28	**88.1**
64	96.13	**99.01**	68.81	**82.11**
81	92.59	**98.84**	49.74	**71.75**
100	90.67	**97.33**	23.31	**49.87**

Table 1: Experiments with large number of modes

Held-out Digit	Bi-GAN	MEG	Core-set+MEG
1	0.287	0.281	**0.351**
4	0.443	0.401	**0.501**
5	0.514	0.402	**0.518**
7	0.347	0.29	**0.387**
9	0.307	0.342	**0.39**

Table 2: Experiments with Anomaly Detection on MNIST dataset. The Held-out digit represents the digit that was held out of the training set during training and treated as the anomaly class. The numbers reported is the area under the precision-recall curve.

with a standard deviation of 0.05. We use a similar experimental setup as Azadi et al. (2018), where our generator and discriminator are parameterized using 4 ReLU-fully connected networks, and use the standard GAN loss in Eq. 1 and 2. To evaluate the performance of the models, we generate $10,000$ samples and assign them to their closest mode. Similar to Azadi et al. (2018), the metrics we use to evaluate performance are: $i)$ 'high quality' samples are samples within 4 standard deviations of the assigned mode and $ii)$ a 'recovered mode' is a mode with at least one assigned sample.

Our results are present in table 1, where we experiment with an increasing number of modes. We see that as the number of modes increase, a normal GAN suffers with increasing mode dropping and lower sample quality compared to Core-set selection. Both observations suggest that using Core-set selection helps to reduce mode-dropping, while generating better samples. Even when using Core-sets for a fewer number of modes, the sampling method is able to outperform a vanilla GAN which suggests that Core-set can outperform random sampling on simple and difficult target distributions. Better sample generation and reduced mode dropping demonstrates the effectiveness of Core-set sampling in maximizing the effectiveness of each batch of data.

4.3 ANOMALY DETECTION

Kumar et al. (2019) proposed using Maximum Entropy generators (MEG) as a generative model which uses an energy function and maximizes the generator's output entropy. They tested their method on various tasks, including anomaly detection for which they reported results that were comparable to the state of the art using GANs (Zenati et al., 2018). Anomaly detection is an important problem in machine learning (Kwon et al., 2017). In anomaly detection, one aims to find samples that are 'anomalies' compared to the training set. Kumar et al. (2019) proposed to use energy estimates from MEG to perform anomaly detection. We guessed that their model could be improved using Core-set selection: Core-set selection allows for better energy estimates made by the generator, since more modes of data will be sampled at each iteration and the model will get to see a distribution that is closer to the true distribution.

We follow their experimental setup by training the generative model with all samples from a chosen MNIST digit left-out during training. Those samples then serve as the 'anomaly class' during evaluation. We report the area under the precision-recall curve and average the score over the last 10 epochs. The results are reported in Table 2 and clearly suggest that MEG greatly benefits from Core-set selection as their method is able to outperform their benchmark. By performing experiments on anomaly detection, we aim to show the generality of the algorithm proposed and not to suggest that MEG is superior to BiGANs on the task. We note that similar improvements may also be ex-

GAN (batch-size = 128)	Small-GAN (batch-size = 128)	GAN (batch-size = 256)	Small-GAN (batch-size = 256)	GAN (batch-size = 512)	Small-GAN (batch-size = 512)
18.75 ± 0.2	**16.73 ± 0.1**	17.9 ± 0.1	**16.22 ± 0.3**	15.68 ± 0.2	**15.08 ± 0.1**

Table 3: FID scores for CIFAR using SN-GAN as the batch-size is exponentially increased. The FID score is calculated using $50,000$ generated samples from the generator.

Small-GAN (batch-size = 64)	GAN (batch-size = 64)	GAN (batch-size = 128)	GAN (batch-size = 256)
13.08	14.82	13.02	12.63

Table 4: FID scores for LSUN using SAGAN as the batch-size is exponentially increased. The FID score is calculated using $50,000$ generated samples from the generator. All experiments were run on the 'outdoor church' subset of the dataset.

pected if Core-set selection is performed with BiGANs. We chose to experiment with MEG since the applicability of our technique outside of generative modeling speaks to its general usefulness.

4.4 IMAGE SYNTHESIS

We also conduct experiments on standard image synthesis benchmarks. To further show the generality of our method, we experiment with two different GAN architectures and two image datasets. We use Spectral Normalization-GAN (Miyato et al., 2018) and Self Attention-GAN (Zhang et al., 2018) on the CIFAR (Krizhevsky et al., 2009) and LSUN (Yu et al., 2015) datasets, respectively. For the LSUN dataset, which consists of 10 different categories, we train the model using the 'outdoor church' subset of the data.

For evaluation, we measured the FID scores (Heusel et al., 2017) of $50,000$ generated samples from the trained models[4]. We compare the performance using SN-GANs with and without Core-set selection across exponentially increasing batch sizes. We observe a similar effect to Brock et al. (2018): just by increasing the mini-batch size by a factor of 4, from 128 to 512, we are able to improve the FID scores from 18.75 to 15.68 for SN-GANs. This further demonstrates the importance of large mini-batches for GAN training. Adding Core-set selection significantly improves the performance of the underlying GAN for all batch-sizes. For a batch size of 128, our model using Core-set sampling significantly outperforms the normal SN-GAN trained with a batch size of 256, and is comparable to an SN-GAN trained with a batch size of 512. The results suggest that the models perform significantly better for any given batch size when Coreset-sampling is used.

However, Core-set sampling does become less helpful as the underlying batch size increases: for SN-GAN, the performance improvement at a batch size of 128 is much larger than the improvement at a batch size of 512. This supports the hypothesis that Core-set selection works by approximating the coverage of a larger batch; a larger batch can already recover more modes of the data - so under this hypothesis, we would expect Core-set selection to help less.

We see similar results when experimenting with Self Attention GANs (SAGAN) (Zhang et al., 2018) on the LSUN dataset (Yu et al., 2015). Compared to our results with SN-GAN, increasing the batch size results in a smaller difference in the performance for the SAGAN model, but we still see the FID improve from 14.82 to 12.63 as the batch-size is increased by a factor of 4. Using Core-set sampling with a batch size of 64, we are able to achieve a comparable score to when the model is trained with a batch size of 128. We believe that one reason for a comparably smaller advantage of using Core-set sampling on LSUN is the nature of the data itself: using the 'outdoor church' subset of LSUN reduces the total number of modes *possible* in the target distribution, since images of churches have fewer differences than the images in CIFAR-10 data set. We see similar effects in the mixture of Gaussians experiment (See 4.2) where the relative difference between a GAN trained with and without Core-set sampling increases as the number of modes are increased.

---

[4] Note that we measure the performance of all the models using the PyTorch version of FID scores, and not the official Tensorflow one. We ran all our experiments with the same code for accurate comparison.

Small-GAN (batch size = 128)	SN-GAN (batch size = 128)	SN-GAN (batch size = 256)	SN-GAN (batch size = 512)
14.51	13.31	26.46	51.64

Table 5: Timing to perform 50 gradient updates for SN-GAN with and without Core-sets. The time is measured in seconds. All the experiments were performed on a single NVIDIA Titan-XP GPU. The sampling factor was 4 for the prior and 8 for the target distribution.

Small-GAN	A	B	C	D	E
16.73	18.75	18.09	17.03	17.88	17.45

Table 6: FID scores for CIFAR using SN-GAN. The experiment list is: A = Training an SN-GAN, B = Core-set selection directly on the images, C = Core-set applied directly on Inception embeddings without a random projection, D = Core-set applied only on the prior distribution, E = Core-set applied only on target distribution.

4.5 TIMING ANALYSIS

Since random sampling can be done very quickly, it is important to investigate the amount of time it takes to train GANs with and without Core-set sampling. We measured the time for SN-GAN to do 50 gradient steps on the CIFAR dataset with various mini-batch sizes: the results are in Table 5. On average, for each gradient step, the time added by performing Core-Set sampling is only 0.024 seconds.

4.6 ABLATION STUDY

We conduct an ablation study to investigate the reasons for the effectiveness of Core-set selection. We also investigate the effect of different sampling factors and other hyper-parameters. We run all ablation experiments on the task of image synthesis using SN-GAN (Miyato et al., 2018) with the CIFAR-10 dataset (Krizhevsky et al., 2009). We use the same hyperparameters as in our main image synthesis experiments and a batch size of 128, unless otherwise stated.

Examination of Main Hyper-Parameters: We examine $i)$ the importance of the chosen target distribution for Core-set selection and $ii)$ the importance of performing Core-set on that target distribution. The FID scores are reported in Table 6.

The importance of the target distribution is clear, since performing Core-set selection directly on the images (experiment B) performs similar to random-sampling. Experiment C supports our hypothesis that performing a random projection on the Inception embeddings can preserve semantic information while reducing the dimensionality of the features. This increases the effectiveness of Core-set sampling and reduces sampling time.

Our ablation study also shows the importance of performing Core-set selection on both the prior and target distribution. The FID scores of the models are considerably lower when Core-set sampling is used on either distribution alone.

Examination of Sampling Factors: Another important hyper-parameter for training GANs using Core-set selection is the sampling factor. In Table 7 we varied the factors by which both the prior and the target distributions were over-sampled. We see that using 4 for the sampling factor for the prior and 8 for the sampling factor for the target distribution results in the best performance.

5 RELATED WORK

5.1 VARIANCE REDUCTION IN GANS

Researchers have proposed reducing variance in GAN training from an optimization perspective, by directly changing the way each of the networks are optimized. Some have proposed applying the extragradient method (Chavdarova et al., 2019), and others have proposed casting the *minimax* two-

A	B	C	D	E	F	G	H	I
18.01	17.8	17.59	17.12	16.83	**16.73**	16.9	17.95	20.79

Table 7: FID scores for CIFAR using SN-GAN. Each of the experiment shows a different pair of over-sampling factors for the prior and target distributions. The factors are listed as: sampling factor for prior distribution × sampling factor for target distribution. A = 2×2; B = 2×4; C = 4×2; D = 4×4; E = 8×4; F = 4×8; G = 8×8; H = 16×16; I = 32×32

player game as a variational-inequality problem (Gidel et al., 2018). Brock et al. (2018) recently proposed to reduce variance directly by using large mini-batch sizes.

5.2 STABILITY IN GAN TRAINING

Stabilizing GANs has been extensively studied theoretically. Researchers have worked on improving the dynamics of the two player minimax game in a variety of ways (Nagarajan & Kolter, 2017; Mescheder et al., 2018; Mescheder, 2018; Li et al., 2017b; Arora et al., 2017). Training instability has been linked to the architectural properties of GANs: specially to the discriminator (Miyato et al., 2018). Proposed architectural stabilization techniques include using Convolutional Neural Networks (CNNs) (Radford et al., 2015), using very large batch sizes (Brock et al., 2018), using an ensemble of the discriminators (Durugkar et al., 2016), using spectral normalization for the discriminator (Miyato et al., 2018), adding self-attention layers for the generator and discriminator networks (Vaswani et al., 2017; Zhang et al., 2018) and using iterative updates to a *global* generator and discriminator using an ensemble of paired generators and discriminators (Chavdarova & Fleuret, 2018). Different objectives have also been proposed to stabilize GAN training (Arjovsky et al., 2017; Gulrajani et al., 2017; Li et al., 2017a; Mao et al., 2017; Mroueh & Sercu, 2017; Bellemare et al., 2017).

5.3 CORE-SET SELECTION

Core-set sampling has been widely studied from an algorithmic perspective in attempts to find better approximate solutions to the original NP-Hard problem (Agarwal et al., 2005; Clarkson, 2010; Pratap & Sen, 2018). The optimality of the sub-sampled solutions have also been studied theoretically (Barahona & Chudak, 2005; Goldman, 1971). See Phillips (2016) for a recent survey on Core-set selection algorithms. Core-sets have been applied to many machine learning problems such as k-means and approximate clustering (Har-Peled & Mazumdar, 2004; Har-Peled & Kushal, 2007; Bādoiu et al., 2002)), active learning for SVMs (Tsang et al., 2005; 2007), unsupervised subset selection for hidden Markov models (Wei et al., 2013) scalable Bayesian inference, (Huggins et al., 2016) and mixture models (Feldman et al., 2011). We are not aware of Core-set selection being applied to GANs.

5.4 CORE-SET SELECTION IN DEEP LEARNING

Core-set selection is largely underexplored in the Deep Learning literature, but interest has recently increased. Sener & Savarese (2017) proposed to use Core-set sampling as a batch-mode active learning sampler for CNNs. Their method used the embeddings of a trained network to sample from. Mussay et al. (2019) proposed using Core-set selection on the activations of a neural network for network compression. Core-set selection has also been used in continual learning to sample points for episodic memory (Nguyen et al., 2017).

6 CONCLUSION

In this work we present a general way to mimic using a large batch-size in GANs while minimizing computational overhead. This technique uses Core-set selection and improves performance in a wide variety of contexts. This work also suggets further research: a similar method could be applied to other learning tasks where large mini-batches may be useful.

REFERENCES

Pankaj K Agarwal, Sariel Har-Peled, and Kasturi R Varadarajan. Geometric approximation via coresets. *Combinatorial and computational geometry*, 52:1–30, 2005.

Martin Arjovsky, Soumith Chintala, and Léon Bottou. Wasserstein gan. *arXiv preprint arXiv:1701.07875*, 2017.

Sanjeev Arora, Rong Ge, Yingyu Liang, Tengyu Ma, and Yi Zhang. Generalization and equilibrium in generative adversarial nets (gans). In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pp. 224–232. JMLR. org, 2017.

Sanjeev Arora, Andrej Risteski, and Yi Zhang. Do gans learn the distribution? some theory and empirics. 2018.

Samaneh Azadi, Catherine Olsson, Trevor Darrell, Ian Goodfellow, and Augustus Odena. Discriminator rejection sampling. *arXiv preprint arXiv:1810.06758*, 2018.

Mihai Bādoiu, Sariel Har-Peled, and Piotr Indyk. Approximate clustering via core-sets. In *Proceedings of the thiry-fourth annual ACM symposium on Theory of computing*, pp. 250–257. ACM, 2002.

Francisco Barahona and FabiáN A Chudak. Near-optimal solutions to large-scale facility location problems. *Discrete Optimization*, 2(1):35–50, 2005.

Marc G Bellemare, Ivo Danihelka, Will Dabney, Shakir Mohamed, Balaji Lakshminarayanan, Stephan Hoyer, and Rémi Munos. The cramer distance as a solution to biased wasserstein gradients. *arXiv preprint arXiv:1705.10743*, 2017.

Andrew Brock, Jeff Donahue, and Karen Simonyan. Large scale gan training for high fidelity natural image synthesis. *arXiv preprint arXiv:1809.11096*, 2018.

Tatjana Chavdarova and François Fleuret. Sgan: An alternative training of generative adversarial networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 9407–9415, 2018.

Tatjana Chavdarova, Gauthier Gidel, François Fleuret, and Simon Lacoste-Julien. Reducing noise in gan training with variance reduced extragradient. *arXiv preprint arXiv:1904.08598*, 2019.

Kenneth L Clarkson. Coresets, sparse greedy approximation, and the frank-wolfe algorithm. *ACM Transactions on Algorithms (TALG)*, 6(4):63, 2010.

Sanjoy Dasgupta and Anupam Gupta. An elementary proof of a theorem of johnson and lindenstrauss. *Random Structures & Algorithms*, 22(1):60–65, 2003.

David L Donoho et al. High-dimensional data analysis: The curses and blessings of dimensionality. *AMS math challenges lecture*, 1(2000):32, 2000.

Ishan Durugkar, Ian Gemp, and Sridhar Mahadevan. Generative multi-adversarial networks. *arXiv preprint arXiv:1611.01673*, 2016.

Ahmet M Eskicioglu and Paul S Fisher. Image quality measures and their performance. *IEEE Transactions on communications*, 43(12):2959–2965, 1995.

Reza Zanjirani Farahani and Masoud Hekmatfar. *Facility location: concepts, models, algorithms and case studies*. Springer, 2009.

William Fedus, Ian Goodfellow, and Andrew M Dai. Maskgan: better text generation via filling in the_. *arXiv preprint arXiv:1801.07736*, 2018.

Dan Feldman, Matthew Faulkner, and Andreas Krause. Scalable training of mixture models via coresets. In *Advances in neural information processing systems*, pp. 2142–2150, 2011.

Gauthier Gidel, Hugo Berard, Gaëtan Vignoud, Pascal Vincent, and Simon Lacoste-Julien. A variational inequality perspective on generative adversarial networks. *arXiv preprint arXiv:1802.10551*, 2018.

Bernd Girod. What's wrong with mean-squared error? *Digital images and human vision*, pp. 207–220, 1993.

AJ Goldman. Optimal center location in simple networks. *Transportation science*, 5(2):212–221, 1971.

Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Advances in neural information processing systems*, pp. 2672–2680, 2014.

Priya Goyal, Piotr Dollár, Ross Girshick, Pieter Noordhuis, Lukasz Wesolowski, Aapo Kyrola, Andrew Tulloch, Yangqing Jia, and Kaiming He. Accurate, large minibatch sgd: Training imagenet in 1 hour. *arXiv preprint arXiv:1706.02677*, 2017.

Ishaan Gulrajani, Faruk Ahmed, Martin Arjovsky, Vincent Dumoulin, and Aaron C Courville. Improved training of wasserstein gans. In *Advances in neural information processing systems*, pp. 5767–5777, 2017.

Jiaxian Guo, Sidi Lu, Han Cai, Weinan Zhang, Yong Yu, and Jun Wang. Long text generation via adversarial training with leaked information. In *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

Sariel Har-Peled and Akash Kushal. Smaller coresets for k-median and k-means clustering. *Discrete & Computational Geometry*, 37(1):3–19, 2007.

Sariel Har-Peled and Soham Mazumdar. On coresets for k-means and k-median clustering. In *Proceedings of the thirty-sixth annual ACM symposium on Theory of computing*, pp. 291–300. ACM, 2004.

Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. Gans trained by a two time-scale update rule converge to a local nash equilibrium. In *Advances in Neural Information Processing Systems*, pp. 6626–6637, 2017.

Jonathan Huggins, Trevor Campbell, and Tamara Broderick. Coresets for scalable bayesian logistic regression. In *Advances in Neural Information Processing Systems*, pp. 4080–4088, 2016.

Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A Efros. Image-to-image translation with conditional adversarial networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 1125–1134, 2017.

Nitish Shirish Keskar, Dheevatsa Mudigere, Jorge Nocedal, Mikhail Smelyanskiy, and Ping Tak Peter Tang. On large-batch training for deep learning: Generalization gap and sharp minima. *arXiv preprint arXiv:1609.04836*, 2016.

Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. Technical report, Citeseer, 2009.

Rithesh Kumar, Anirudh Goyal, Aaron Courville, and Yoshua Bengio. Maximum entropy generators for energy-based models. *arXiv preprint arXiv:1901.08508*, 2019.

Donghwoon Kwon, Hyunjoo Kim, Jinoh Kim, Sang C Suh, Ikkyun Kim, and Kuinam J Kim. A survey of deep learning-based network anomaly detection. *Cluster Computing*, pp. 1–13, 2017.

Christian Ledig, Lucas Theis, Ferenc Huszár, Jose Caballero, Andrew Cunningham, Alejandro Acosta, Andrew Aitken, Alykhan Tejani, Johannes Totz, Zehan Wang, et al. Photo-realistic single image super-resolution using a generative adversarial network. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 4681–4690, 2017.

Chun-Liang Li, Wei-Cheng Chang, Yu Cheng, Yiming Yang, and Barnabás Póczos. Mmd gan: Towards deeper understanding of moment matching network. In *Advances in Neural Information Processing Systems*, pp. 2203–2213, 2017a.

Jerry Li, Aleksander Madry, John Peebles, and Ludwig Schmidt. Towards understanding the dynamics of generative adversarial networks. *arXiv preprint arXiv:1706.09884*, 2017b.

Xudong Mao, Qing Li, Haoran Xie, Raymond YK Lau, Zhen Wang, and Stephen Paul Smolley. Least squares generative adversarial networks. In *Proceedings of the IEEE International Conference on Computer Vision*, pp. 2794–2802, 2017.

Lars Mescheder. On the convergence properties of gan training. *arXiv preprint arXiv:1801.04406*, 1:16, 2018.

Lars Mescheder, Andreas Geiger, and Sebastian Nowozin. Which training methods for gans do actually converge? *arXiv preprint arXiv:1801.04406*, 2018.

Takeru Miyato, Toshiki Kataoka, Masanori Koyama, and Yuichi Yoshida. Spectral normalization for generative adversarial networks. *arXiv preprint arXiv:1802.05957*, 2018.

Youssef Mroueh and Tom Sercu. Fisher gan. In *Advances in Neural Information Processing Systems*, pp. 2513–2523, 2017.

Ben Mussay, Samson Zhou, Vladimir Braverman, and Dan Feldman. On activation function coresets for network pruning. *arXiv preprint arXiv:1907.04018*, 2019.

Vaishnavh Nagarajan and J Zico Kolter. Gradient descent gan optimization is locally stable. In *Advances in Neural Information Processing Systems*, pp. 5585–5595, 2017.

Cuong V Nguyen, Yingzhen Li, Thang D Bui, and Richard E Turner. Variational continual learning. *arXiv preprint arXiv:1710.10628*, 2017.

Jeff M Phillips. Coresets and sketches. *arXiv preprint arXiv:1601.00617*, 2016.

Rameshwar Pratap and Sandeep Sen. Faster coreset construction for projective clustering via low-rank approximation. In *International Workshop on Combinatorial Algorithms*, pp. 336–348. Springer, 2018.

Alec Radford, Luke Metz, and Soumith Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks. *arXiv preprint arXiv:1511.06434*, 2015.

Tim Salimans, Han Zhang, Alec Radford, and Dimitris Metaxas. Improving gans using optimal transport. *arXiv preprint arXiv:1803.05573*, 2018.

Ozan Sener and Silvio Savarese. Active learning for convolutional neural networks: A core-set approach. *arXiv preprint arXiv:1708.00489*, 2017.

Christopher J Shallue, Jaehoon Lee, Joe Antognini, Jascha Sohl-Dickstein, Roy Frostig, and George E Dahl. Measuring the effects of data parallelism on neural network training. *arXiv preprint arXiv:1811.03600*, 2018.

Samarth Sinha, Sayna Ebrahimi, and Trevor Darrell. Variational adversarial active learning. *arXiv preprint arXiv:1904.00370*, 2019.

Samuel L Smith, Pieter-Jan Kindermans, Chris Ying, and Quoc V Le. Don't decay the learning rate, increase the batch size. *arXiv preprint arXiv:1711.00489*, 2017.

Christian Szegedy, Sergey Ioffe, Vincent Vanhoucke, and Alexander A Alemi. Inception-v4, inception-resnet and the impact of residual connections on learning. In *Thirty-First AAAI Conference on Artificial Intelligence*, 2017.

Ivor W Tsang, James T Kwok, and Pak-Ming Cheung. Core vector machines: Fast svm training on very large data sets. *Journal of Machine Learning Research*, 6(Apr):363–392, 2005.

Ivor W Tsang, Andras Kocsor, and James T Kwok. Simpler core vector machines with enclosing balls. In *Proceedings of the 24th international conference on Machine learning*, pp. 911–918. ACM, 2007.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in neural information processing systems*, pp. 5998–6008, 2017.

Kai Wei, Yuzong Liu, Katrin Kirchhoff, and Jeff Bilmes. Using document summarization techniques for speech data subset selection. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 721–726, 2013.

Laurence A Wolsey and George L Nemhauser. *Integer and combinatorial optimization*. John Wiley & Sons, 2014.

Yongqin Xian, Tobias Lorenz, Bernt Schiele, and Zeynep Akata. Feature generating networks for zero-shot learning. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 5542–5551, 2018.

Jason Yosinski, Jeff Clune, Yoshua Bengio, and Hod Lipson. How transferable are features in deep neural networks? In *Advances in neural information processing systems*, pp. 3320–3328, 2014.

Fisher Yu, Ari Seff, Yinda Zhang, Shuran Song, Thomas Funkhouser, and Jianxiong Xiao. Lsun: Construction of a large-scale image dataset using deep learning with humans in the loop. *arXiv preprint arXiv:1506.03365*, 2015.

Houssam Zenati, Chuan Sheng Foo, Bruno Lecouat, Gaurav Manek, and Vijay Ramaseshan Chandrasekhar. Efficient gan-based anomaly detection. *arXiv preprint arXiv:1802.06222*, 2018.

Han Zhang, Tao Xu, Hongsheng Li, Shaoting Zhang, Xiaogang Wang, Xiaolei Huang, and Dimitris N Metaxas. Stackgan: Text to photo-realistic image synthesis with stacked generative adversarial networks. In *Proceedings of the IEEE International Conference on Computer Vision*, pp. 5907–5915, 2017.

Han Zhang, Ian Goodfellow, Dimitris Metaxas, and Augustus Odena. Self-attention generative adversarial networks. *arXiv preprint arXiv:1805.08318*, 2018.

Jia-Jie Zhu and José Bento. Generative adversarial active learning. *arXiv preprint arXiv:1702.07956*, 2017.

Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *Proceedings of the IEEE international conference on computer vision*, pp. 2223–2232, 2017.