Active Mini-Batch Sampling Using Repulsive Point Processes

Cheng Zhang,[1] **Cengiz Öztireli,**[2] **Stephan Mandt,**[3] **Giampiero Salvi**[4]

1. Microsoft Research, Cambridge, UK, Cheng.Zhang@microsoft.com
2. Disney Research, Zurich, Switzerland, cengiz.oztireli@disneyresearch.com
3. University of California,Irvine,Los Angeles, USA, stephan.mandt@gmail.com
4. KTH Royal Institute of Technology, Stockholm, Sweden, giampi@kth.se

Abstract

The convergence speed of stochastic gradient descent (SGD) can be improved by actively selecting mini-batches. We explore sampling schemes where similar data points are less likely to be selected in the same mini-batch. In particular, we prove that such repulsive sampling schemes lower the variance of the gradient estimator. This generalizes recent work on using Determinantal Point Processes (DPPs) for mini-batch diversification (Zhang et al., 2017) to the broader class of repulsive point processes. We first show that the phenomenon of variance reduction by diversified sampling generalizes in particular to non-stationary point processes. We then show that other point processes may be computationally much more efficient than DPPs. In particular, we propose and investigate Poisson Disk sampling—frequently encountered in the computer graphics community—for this task. We show empirically that our approach improves over standard SGD both in terms of convergence speed as well as final model performance.

Introduction

Stochastic gradient descent (SGD) (Bottou 2010) is key to modern scalable machine learning. Combined with back-propagation, it forms the foundation to train deep neural networks (LeCun et al. 1998b). Applied to variational inference (Hoffman et al. 2013; Zhang et al. 2017a), it enables the use of probabilistic graphical models on massive data. SGD training has contributed to breakthroughs in many applications (Krizhevsky et al. 2012; Mikolov et al. 2013).

A key limitation for the speed of convergence of SGD algorithms is the stochastic gradient noise. Smaller gradient noise allows for larger learning rates and therefore faster convergence. For this reason, variance reduction for SGD is an active research topic.

Several recent works have shown that the variance of SGD can be reduced when diversifying the data points in a mini-batch based on their features (Zhao and Zhang 2014; Fu and Zhang 2017; Zhang et al. 2017b). When data points are coming from similar regions in feature space, their gradient contributions are positively correlated. Diversifying data

points by sampling from different regions in feature space de-correlates their gradient contributions, which leads to a better gradient estimation.

Another benefit of actively biasing the mini-batch sampling procedure relates to better model performance (Chang et al. 2017; Shrivastava et al. 2016; Zhang et al. 2017b). Zhang et al. (2017b) biased the data towards a more uniform distribution, upsampling data-points in scarce regions and downsampling data points in dense regions, leading to a better performance during test time. Chen and Gupta (2015) showed that training on simple classification tasks first, and later adding more difficult examples, leads to a clear performance gain compared to training on all examples simultaneously. We refer to such schemes which modify the marginal probabilities of each selected data point as *active bias*. The above results suggest that utilizing an active bias in mini-batch sampling can result in improved performance without additional computational cost.

In this work, we present a framework for active mini-batch sampling based on repulsive point processes. The idea is simple: we specify a data selection mechanism that actively selects a mini-batch based on features of the data. This mechanism introduces *repulsion* between the data points, meaning that it suppresses data points with similar features to co-occur in the same mini-batch. We use repulsive point processes for this task. Finally, the chosen mini-batch is used to perform a stochastic gradient step, and this scheme is repeated until convergence.

Our framework generalizes the recently proposed mini-batch diversification based on determinantal point processes (DPP) (Zhang et al. 2017b) to a much broader class of repulsive processes, and allows users to encode any preferred active bias with efficient mini-batch sampling algorithms. In more detail, our contributions are as follows:

1. *We propose to use point processes for active mini-batch selection.*
   We provide a theoretical analysis which shows that mini-batch selection with repulsive point processes may reduce the variance of stochastic gradient descent. The proposed approach can accommodate point processes with adaptive densities and adaptive pair-wise correlations. Thus, we can use it for data subsampling with an active bias.

2. *Going beyond DPPs, we propose a group of more efficient*

repulsive point processes based on Poisson Disk Sampling (PDS).

We propose PDS with dart throwing for mini-batch selection. Compared to DPPs, this improves the sampling costs from Nk^3 to merely k^2, where N is the number of data points and k is the mini-batch size.

We propose a dart-throwing method with an adaptive disk size and adaptive densities to sample mini-batches with an active bias.

3. *We test our proposed method on several machine learning applications from the domains of computer vision and speech recognition. We find increased model performance and faster convergence due to variance reduction.*

Related Work

In this section, we begin with reviewing the most relevant work on *diversified mini-batch sampling*. Then, we discuss the benefits of subsampling schemes with an *active bias*, where data are either reweighed or re-ordered. Finally, we review the relevant aspects of *point processes*.

Diversified Mini-Batch Sampling   Prior research (Zhao and Zhang 2014; Fu and Zhang 2017; Zhang et al. 2017b; Yin et al. 2017) has shown that sampling diversified mini-batches can reduce the variance of stochastic gradients. It is also the key to overcome the problem of the saturation of the convergence speed in the distributed setting (Yin et al. 2017). Diversifying the data is also computationally efficient for large-scale learning problems (Zhao and Zhang 2014; Fu and Zhang 2017; Zhang et al. 2017b). Buchholz et al. (2018) used diversified sampling for optimizing Monte-Carlo objectives in variational inference.

Zhang et al. (2017b) recently proposed to use DPPs for diversified mini-batch sampling and drew the connection to stratified sampling (Zhao and Zhang 2014) and clustering-based preprocessing for SGD (Fu and Zhang 2017). A disadvantage of the DPP-approach is the computational overhead. Besides presenting a more general theory, we provide more efficient point processes in this work.

Active Bias   Different types of active bias in subsampling the data can improve the convergence and lead to improved final performance in model training (Alain et al. 2015; Gopal 2016; Chang et al. 2017; Chen and Gupta 2015; Shrivastava et al. 2016). As summarized in Chang et al. (2017), self-paced learning biases towards easy examples in the early learning phase. Active-learning, on the other hand, puts more emphasis on uncertain cases, and hard example mining focuses on the difficult-to-classify examples.

Chang et al. (2017) investigate a supervised setup and sample data points, which have high prediction variance, more often. (Gopal 2016) (2017) maintain a desired class distribution during mini-batch sampling. Diversified mini-batch sampling with DPPs (Zhang et al. 2017b) re-weights the data towards a more uniform distribution, which improves the final performance when the data set is imbalanced.

The choice of a good active bias depends on the data set and problem at hand. Our proposed method is compatible with different active bias preferences.

Point Processes   Point processes have a long history in physics and mathematics, and are heavily used in computer graphics (Macchi 1975; Ripley 1976; Illian et al. 2008; Öztireli and Gross 2012; Lavancier et al. 2015). DPPs, as a group of point processes, have been introduced and used in the machine learning community in recent years (Kulesza et al. 2012; Li et al. 2015; Kathuria et al. 2016).

Other types of point processes have been less explored and used in machine learning. There are many different repulsive point processes, such as PDS, or Gibbs processes, with properties similar to DPPs, but with significantly higher sampling efficiency (Illian et al. 2008). Additionally, more flexible point processes with adaptive densities and interactions are well studied in computer graphics (Li et al. 2010; Roveri et al. 2017; Kita and Miyata 2016), but not explored much in the machine learning community. Our proposed framework is based on generic point processes. As one of the most efficient repulsive point processes, we advocate Poisson disk sampling in this paper.

Repulsive Point Processes for Variance Reduction

In this section, we first briefly introduce our main idea of using point processes for mini-batch sampling in the context of the problem setting and revisit point processes. We prove that any repulsive point process can lead to reduced gradient variance in SGD, and discuss the implications of this result. The theoretical analysis in this section leads to multiple practical algorithms.

Problem Setting

Consider a loss function $\ell(\mathbf{x}, \theta)$, where θ are the model parameters, and $\mathbf{x}$ indicates the data. In this paper, we consider a modified empirical risk minimization problem (Zhang et al. 2017b):

$$\hat{J}(\theta) = \mathbb{E}_{\hat{\mathbf{x}} \sim \mathcal{P}}[\ell(\hat{\mathbf{x}}, \theta)]. \tag{1}$$

$\mathcal{P}$ indicates a point process defining a distribution over subsets $\hat{\mathbf{x}}$ of the data, which will be specified below. Note that this leads to a potentially biased version of the standard empirical risk (Bottou 2010). Debiasing the objective is possible. It requires re-weighting the loss for each data point inverse proportional to its marginal probability to be sampled (Zhang et al. 2017b).

We optimize Eq. 1 via stochastic gradient descent, which leads to the updates

$$\theta_{t+1} = \theta_t - \rho_t \hat{G} = \theta_t - \rho_t \frac{1}{|B|} \sum_{i \in B} \nabla \ell(x_i, \theta), \quad B \sim \mathcal{P}.$$

$B \subset \{1, \ldots, N\}$, a set of data indices that define the mini-batch. $\hat{G}$ is the gradient estimated from a mini-batch. The data points chosen for each mini-batch are drawn from a point process $\mathcal{P}$, which defines probability measures over
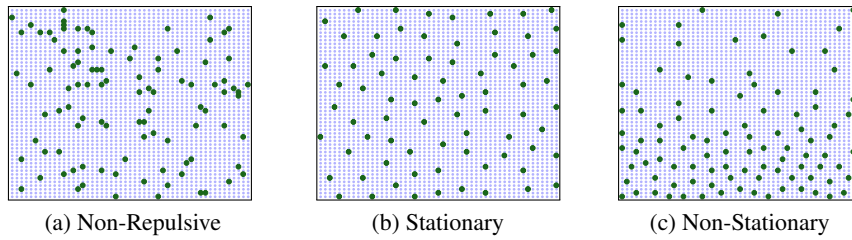
Figure 1: Examples of sampling a subset of data points. We sampled different subsets of 100 points (dark green) from a bigger set of points (light blue), using three different point processes. Panel 1(a) shows a uniformly randomly sampled subset. Panel 1(b) and panel 1(c) show two examples with different repulsive point processes.

different mini-batches. Therefore, our scheme generalizes SGD in that the data points in the mini-batch B are selected actively, rather than uniformly.

Figure 1 shows examples of subset selection using different point processes. Drawing the data randomly without replacement corresponds to a point process as well, thus standard SGD trivially belongs to the class of algorithms considered here. In this paper, we investigate different point processes and analyze how they improve the performance of different models on empirical data sets.

Background on Point Processes

Point processes are generative processes of collections of points in some measure space (Møller and Waagepetersen 2004; Illian et al. 2008). They can be used to sample subsets of data with various properties, either from continuous spaces or from discrete sets, such as a finite dataset. In this paper, we explore different point processes to sample mini-batches with different statistical properties.

More formally, a point process $\mathcal{P}$ in $\mathbb{R}^d$ can be defined by considering the joint probabilities of finding points generated by this process in infinitesimal volumes. One way to express these probabilities is via *product densities*. Let $\mathbf{x}_i$ denote some arbitrary points in $\mathbb{R}^d$, and $\mathcal{B}_i$ infinitesimal spheres centered at these points with volumes $d\mathbf{x}_i = |\mathcal{B}_i|$. Then the nth order product density $\varrho^{(n)}$ is defined by

$$p(\mathbf{x}_1, \cdots, \mathbf{x}_n) = \varrho^{(n)}(\mathbf{x}_1, \cdots, \mathbf{x}_n)d\mathbf{x}_1 \cdots d\mathbf{x}_n,$$

where $p(\mathbf{x}_1, \cdots, \mathbf{x}_n)$ is the joint probability of having a point of the point process $\mathcal{P}$ in each of the infinitesimal spheres $\mathcal{B}_i$. We can use $\mathcal{P}$ to generate infinitely many point configurations, each corresponding to e.g. a mini-batch.

For example, DPP defines this probability of sampling a subset as being proportional to the determinant of a kernel matrix. It is thus described by the nth order product density (Lavancier et al. 2012): $\varrho^{(n)}(\mathbf{x}_1, \cdots, \mathbf{x}_n) = det[C](\mathbf{x}_1, \cdots, \mathbf{x}_n)$, where $det[C](\mathbf{x}_1, \cdots, \mathbf{x}_n)$ is the determinant of the $n \times n$ sized sub-matrix of kernel matrix C with entries specified by $\mathbf{x}_1, \cdots, \mathbf{x}_n$.

For our analysis, we will just need the first and second order product density, which are commonly denoted by $\lambda(\mathbf{x}) := \varrho^{(1)}(\mathbf{x})$, $\varrho(\mathbf{x}, \mathbf{y}) := \varrho^{(2)}(\mathbf{x}, \mathbf{y})$. An important special case of point processes is stationary processes. For such processes, the point distributions generated are translation invariant, where the intensity is a constant.

Point Processes for Active Mini-Batch Sampling

Recently, Zhang et al. (2017b) investigated how to utilize a particular type of point process, DPP, for mini-batch diversification. Here, we generalize the theoretical results to arbitrary stochastic point processes, and elaborate on how the resulting formulations can be utilized for SGD based algorithms. This opens the door to exploiting a vast literature on the theory of point processes, and efficient algorithms for sampling.

SGD-based algorithms utilize the estimator $\hat{\mathbf{G}}(\theta) = \frac{1}{|B|} \sum_{i \in B} \nabla\ell(x_i, \theta)$ for the gradient of the objective. Each mini-batch, i.e. set of data points in this estimator, can be considered as an instance of an underlying point process $\mathcal{P}$. Our goal is to design sampling algorithms for improved learning performance by altering the bias and variance of this gradient estimator.

We first derive a closed form formula for the variance $\text{var}_{\mathcal{P}}(\hat{\mathbf{G}})$ of the gradient estimator for general point processes. We then show that, under mild regularity assumptions, repulsive point processes generally imply variance reduction. For what follows, let $\mathbf{g}(\mathbf{x}, \theta) = \nabla\ell(\mathbf{x}, \theta)$ denote the gradient of the loss function, and recall that $k = |B|$, the mini-batch size.

Theorem 1. *The variance* $\text{var}_{\mathcal{P}}(\hat{\mathbf{G}})$ *of the gradient estimate* $\hat{\mathbf{G}}$ *in SGD for a general stochastic point process* $\mathcal{P}$ *is given by:*

$$\text{var}_{\mathcal{P}}(\hat{\mathbf{G}}) = \frac{1}{k^2} \int_{\mathcal{V} \times \mathcal{V}} \lambda(\mathbf{x})\lambda(\mathbf{y})\mathbf{g}(\mathbf{x}, \theta)^T \mathbf{g}(\mathbf{y}, \theta) \quad (2)$$
$$\left[\frac{\varrho(\mathbf{x}, \mathbf{y})}{\lambda(\mathbf{x})\lambda(\mathbf{y})} - 1 \right] d\mathbf{x}d\mathbf{y}$$
$$+ \frac{1}{k^2} \int_{\mathcal{V}} \|\mathbf{g}(\mathbf{x}, \theta)\|^2 \lambda(\mathbf{x})d\mathbf{x}.$$

Proof. In Appendix. $\square$

Remark 1. This formula applies to general point processes and hence sampling strategies for mini-batches. It proves that variance only depends on first and second order correlations captured by $\lambda(\mathbf{x})$ and $\varrho(\mathbf{x}, \mathbf{y})$, respectively. This provides a simple and convenient tool for analyzing properties of sampling strategies with respect to dataset characteristics for variance control, once only these lower order sampling characteristics are known or estimated by simulation.

Remark 2. For standard SGD, we have $\varrho(\mathbf{x}, \mathbf{y}) = \lambda(\mathbf{x})\lambda(\mathbf{y})$. This is due to the nature of random sampling, where sampling a point is independent of already sampled points. Note that this applies also to adaptive sampling with non-constant $\lambda(\mathbf{x})$. Hence, the term $[\frac{\varrho(\mathbf{x},\mathbf{y})}{\lambda(\mathbf{x})\lambda(\mathbf{y})} - 1]$ vanishes in SGD. In contrast, we show next that this term may induce a variance reduction for repulsive point processes.

Remark 3. Repulsive point processes may make the first term in Eq. 2 negative, implying variance reduction. For repulsive point processes, the probability of sampling points that are close to each other is low. Consequently, if $\mathbf{x}$ and $\mathbf{y}$ are close, $\varrho(\mathbf{x}, \mathbf{y}) < \lambda(\mathbf{x})\lambda(\mathbf{y})$, and the term $[\frac{\varrho(\mathbf{x},\mathbf{y})}{\lambda(\mathbf{x})\lambda(\mathbf{y})} - 1]$ is negative. This is due to points repelling each other (we will elaborate more on this in the next section). Furthermore, assuming that the loss function is sufficiently smooth in its data argument, the gradients are aligned for close points i.e $\mathbf{g}(\mathbf{x}, \theta)^T \mathbf{g}(\mathbf{y}, \theta) > 0$. This combined implies that close points provide negative contributions to the first integral in Eq. 2. The contributions of points farther apart average out and become negligible due to gradients not being correlated with $\varrho(\mathbf{x}, \mathbf{y})$, which is the case for all current sampling algorithms and the ones we propose in the next section. The negative first term in Eq. 2 leads to variance reduction, for repulsive point processes.

Implications. This proposed theory allows us to use any point process for mini-batch sampling, such as DPP, finite Gibbs processes, Poisson disk sampling (PDS), and many more (Illian et al. 2008). It thus offers many new directions of possible improvement. Foremost, we can choose point processes with a different degree of repulsion (Biscio et al. 2016), and computational efficiency. Furthermore, in this general theory, we are able to adapt the density and alter the pair-wise interactions to encode our preference. In the next section, we propose several practical algorithms utilizing these benefits.

Poisson Disk Sampling for Active Mini-batch Sampling

We adapt efficient dart throwing algorithms for fast repulsive and adaptive mini-batch sampling. We further extend the algorithm with an adaptive disk size and density. For supervised setups, we shrink the disc size towards the decision boundary, using mingling indices (Illian et al. 2008). This biases towards hard examples and improves classification accuracy.

Stationary Poisson Disk Sampling

PDS is one type of repulsive point process. It demonstrates stronger local repulsion compared to DPP (Biscio et al. 2016). Typically, it is implemented with the efficient dart throwing algorithm (Lagae and Dutré 2008), and provides similar point arrangements to DPP, albeit much more efficiently.

This process dictates that the smallest distance between each pair of sample points should be at least r with respect to some distance measure D. The second order product density
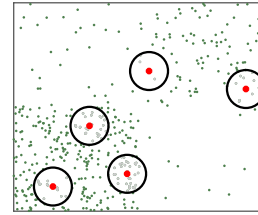

Figure 2: Demonstration of PDS. The black circles of a certain radius r mark regions claimed by the collected points (red). For the next iteration, if the newly sampled point falls in any of the circles (points colored in gray), this point will be rejected.

$\varrho(x, y)$ for PDS is zero when the distance between two points are smaller than the disk radius $\|x - y\| \leq r$, and converges to $\varrho(x, y) = \lambda(x)\lambda(y)$ when the two points are far (Öztireli and Gross 2012). Thus, $\left[\frac{\varrho(\mathbf{x},\mathbf{y})}{\lambda(\mathbf{x})\lambda(\mathbf{y})} - 1\right] = -1 < 0$ when the points are within distance r, and 0 when they are far.

As demonstrated in Figure 2, the basic dart throwing algorithm for PDS works as follows in each iteration: 1) randomly sample a data point; 2) if it is within a distance r of any already accepted data point, reject; otherwise, accept the new point. We can also specify the maximum sampling size k. This means that we terminate the algorithm when k points have been accepted. The computational complexity of PDS with dart throwing[2] is $\mathcal{O}(k^2)$. This is much lower than the complexity $\mathcal{O}(N k^3)$ for k-DPP, where N is the number of data points in the dataset. In the rest of the paper, we refer to this version of PDS as **"Vanilla PDS"**.

Poisson Disk Sampling with Adaptive Density

To further utilize the potential benefit of our framework, we propose several variations of PDS. In particular, we use mingling index based marked processes. We then propose three variants as explained below: Easy PDS, where only the points far from decision boundaries repulse each other, as well as Dense PDS and Anneal PDS, where we can impose preferences on point densities.

Mingling Index The mingling index $\bar{M}_K(x_i)$ is defined as (Illian et al. 2008):

$$\bar{M}_K(x_i) = \frac{1}{K} \sum_{j=1}^{K} \mathbf{1}\big(m(x_i) \neq m(x_j)\big), \qquad (3)$$

where $m(x_i)$ indicates the mark of the point x_i. In case of a classification task, the mark is the class label. $\bar{M}_K(x_i)$ is the ratio of points with different marks than x_i among its K nearest neighbors.

Depending on the mingling index, there are three different situations. Firstly, if $\bar{M}_K(x_i) = 0$, the region around x_i only

---

[2] In practice, the number of accepted points can be smaller than the number of considered points in the dataset, as some of them will be eliminated by not satisfying the distance criteria.

Algorithm 1 Draw throwing for Dense PDS

Input: data x, mini-batch size S, mingling index M for all data, the parameter for categorical distribution to sample mingling index π, disk radius r_0
  repeat
      sample a mingling index $m \sim \text{Cat}(\pi)$
      randomly sample a point i with mingling index m
      if x_i is not in disk of any samples **then**
          insert x_i to B
      end if
  until S points have been accepted for the mini-batch

---

includes points from the same class. This makes x_i a relatively easy point to classify. This type of points is preferred to be sampled in the early iterations for self-paced learning. Secondly, if $\bar{M}_K(x_i) > 0$, this point may be close to a decision boundary. For variance reduction, we do not need to repulse this type of points. Additionally, sampling this type of points more often may help the model to refine the decision boundaries. Finally, if $\bar{M}_K(x_i)$ is very high, the point is more likely to be a hard negative. In this case, the point is mostly surrounded by points from other classes. On a side note, points with high mingling indices can be interpreted as support vectors (Cortes and Vapnik 1995).

Adaptive Variants of Poisson Disk Sampling Gradients may change drastically when points are close to decision boundaries. Points in this region, thus, violate the assumption in Remark 3. Because of this, our first simple extension, which we call **"Easy PDS"**, sets the disk radius to r_0 when the point has a mingling index $M_{x_i} = 0$, and to 0 if $M_{x_i} > 0$. This means that only easy points (with $M_{x_i} = 0$) repulse. On average, "Easy PDS" is expected to sample more of the difficult points compared with "Vanilla PDS".

For many tasks, when the data is highly structured, there are only few data points that are close to the decision boundary. To refine the decision boundary, inspired by hard example mining, we can sample points with a high mingling index more often. We thus propose the **"Dense PDS"** method summarized in Algorithm 1. Instead of drawing darts randomly, we draw darts based on different mingling indices. The mingling indices can assume $K + 1$ values, where K is the number of nearest neighbors. We thus can specify a parameter π for a categorical distribution to sample mingling indices first. Then we randomly sample a dart which has the given mingling index. In this way, we can encode our preferred density with respect to the mingling index.

It is straightforward to introduce an annealing mechanism in "Dense PDS" by using a different π_n at each iteration n. Inspired by self-paced learning, we can give higher density to points with low mingling index in the early iterations, and slowly increase the density of points with high mingling index. We refer this method as **"Anneal PDS"**.

Note that all our proposed sampling methods only rely on the properties of the given data instead of any model parameters. Thus, they can be easily used as a pre-processing step, prior to the training procedure of the specific model.

Experiments

We evaluate the performance of the proposed method in various application scenarios. Our methods show clear improvements compared to baseline methods in each case. We first demonstrate the behavior of different varieties of our method using a synthetic dataset. Secondly, we compare Vanilla PDS with DPP as in (Zhang et al. 2017b) on the Oxford flower classification task. Finally, we evaluate our proposed methods with two different tasks with very different properties: image classification with MNIST, and speech command recognition.

Synthetic Data We evaluate our methods on two-dimensional synthetic datasets to illustrate the behavior of different sampling strategies. Figure 3 shows two classes (green and red dots) separated by a wave-shaped (sine curve) decision boundary (yellow line). Intuitively, it should be favorable to sample diverse subsets and even more beneficial to give more weight to the data points at the decision boundary, i.e., sampling them more often. We sample one mini-batch with batch size 30 using different sampling methods. For each method, we train a neural network classifier with one hidden layer of five units, using a single mini-batch. This model, albeit simple, is sufficient to handle the non-linear decision boundary in the example.

Figure 3 shows the decision boundaries by repeating the experiment 30 times. In order to illustrate the sampling schemes, we also show one example of sampled mini-batch using blue dots. In the random sampling case (Figure 3(a)), we can see that the mini-batch is not a good representation of the original dataset as some random regions are more densely or sparsely sampled. Consequently, the learned decision boundary is very different from the ground truth. Figures 3(b) shows Vanilla PDS. Because of the repulsive effect, the sampled points cover the data space more uniformly and the decision boundary is improved compared to Figure 3(a). In Figure 3(c), we used Easy PDS, where the disk radius adapts with the mingling index of the points. We can see that points close to the decision boundary do not repel each other. This leads to a potentially more refined decision boundary as compared to Figure 3(b). Finally, Dense PDS, shown in Figure 3(d), chooses more samples close to the boundary and leads to the most precise decision boundary with a single mini-batch.

Moreover, to demonstrate the scalibilty of PDS, we generate synthetic datasets with different dataset sizes and sample mini-batches using PDS. We report the sampling time in Table 1. We can see that the sampling time for a mini-batch using PDS is stable across different dataset sizes N and only differs with respect to mini-batch size k. More experiments on variance reduction are shown in the appendix.

Oxford Flower Classification We compare our proposed PDS with DPP for mini-batch sampling on the fine-grained classification task as in (Zhang et al. 2017b) with the same experimental setting. We use Vanilla PDS (with fixed disk radius) for fair comparison with DPP. Figure 4 shows the test accuracy at the end of each training epoch. We see that sampling with either DPP or PDS leads to accelerated convergence as compared to traditional SGD. With similar performance as

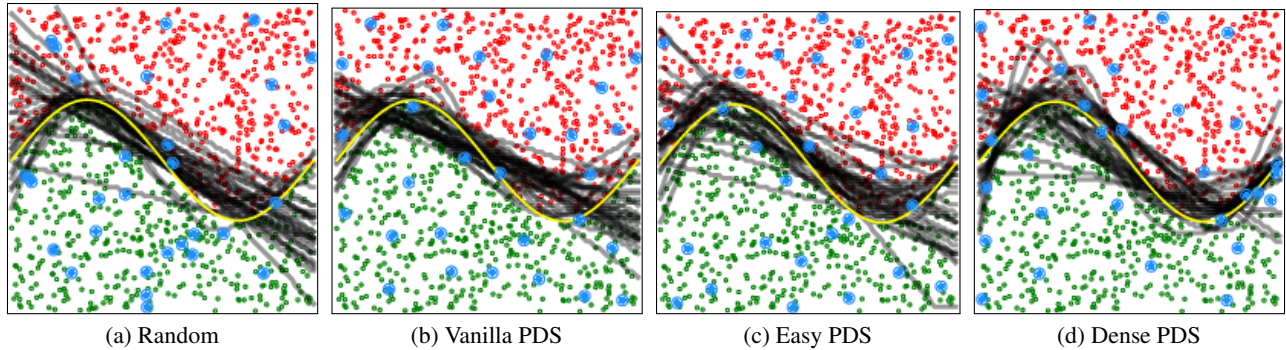| (a) Random | (b) Vanilla PDS | (c) Easy PDS | (d) Dense PDS |

Figure 3: Comparison of performance using one mini-batch sample distribution on synthetic data. Each experiment is repeated 30 times. The resulting decision boundaries are drawn as transparent black lines and the ground truth decision boundary is shown in yellow. As an example, the points from one of the sampled mini-batches are shown in blue.

	N: 4096	N: 8192	N: 16384	N: 32768	N: 65536	N: 131072	N: 262144	N: 524288	N: 1048576
k=50	0.0066	0.0064	0.0064	0.0061	0.0062	0.0069	0.0063	0.0060	0.0061
k=100	0.0239	0.0237	0.0235	0.0234	0.0227	0.0225	0.0228	0.0239	0.0262

Table 1: Average CPU time (sec) to sample each mini-batch using PDS. N is the number of data and k is the mini-batch size. The table confirms that the computational complexity of PDS is $\mathcal{O}(k^2)$, which does not depend on the dataset size N, as opposed to $\mathcal{O}(Nk^3)$ for k-DPP. Thus, PDS is able to scale to massive datasets with large mini-batches. Note that, vanilla PDS is used here, however, other variants of PDS have the same computational complexity for sampling but may have some pre-processing overhead such as computation of mingling index.
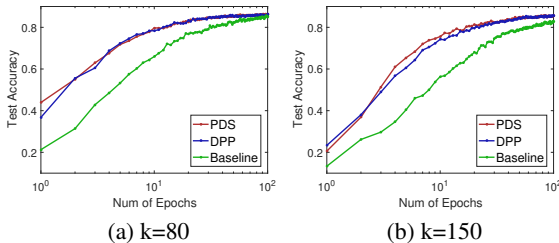

| (a) k=80 | (b) k=150 |

Figure 4: Oxford Flower Classification with Softmax. PDS has similar performance as DPP for sampling mini-batches. However PDS is more efficient as Table 2 shows. Both methods converges faster than traditional SGD (Baseline).

k	50	80	150
k-DPP	7.1680	29.687	189.0303
Fast k-DPP	0.1032	0.3312	1.8745
PDS	0.0461	0.0795	0.1657

Table 2: CPU time (sec) to sample one mini-batch. The disk radius r is set to half of the median value of the distance measure D for PDS.

using DPP, PDS demonstrates significant improvement on sampling efficiency as shown in Table 2. More experimental results with different settings are presented in the appendix.

MNIST We further show results for hand-written digit classification on the MNIST dataset (LeCun et al. 1998a). We compare different variations of our method with two baselines: the traditional SGD and ActiveBias (Chang et al. 2017). We use half of the training data and the full test data. As detailed in the appendix, with MNIST, data are well clustered and most data points have mingling index 0 (easy to classify). A standard multi-layer convolutional neural network (CNN)

from Tensorflow[3] is used in this experiment with standard experimental settings (details in appendix).

Figure 5(a) shows the test error rate evaluated after each SGD training iteration for different mini-batch sampling methods. All active sampling methods with PDS lead to improved final performance compared to traditional SGD. Vanilla PDS clearly outperforms the baseline method. Easy PDS, performs very similarly to Vanilla PDS with slightly faster convergence in the early iterations. Dense PDS leads to better final performance at the cost of a slight decrease in initial convergence speed. The decision boundary is refined because we prefer non-trivial points during training. Anneal PDS further improves Dense PDS with accelerated convergence in the early iterations. Figure 5(b) shows the performance of test accuracy for Anneal PDS compared with baseline methods in a zoomed view.

We thus conclude that all different variations of PDS obtain better final performance, or conversely, achieve the baseline

---

[3]https://www.tensorflow.org/versions/r0.12/tutorials/mnist/pros/

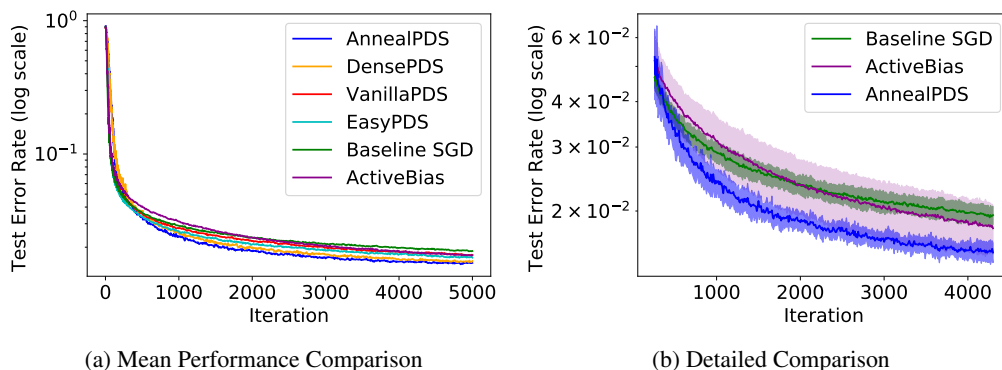(a) Mean Performance Comparison   (b) Detailed Comparison

Figure 5: MNIST experiment (10 repetitions). The mean performance for each method is reported in Panel (a). We compared all variations of our proposed methods with two baselines: Baseline SGD and ActiveBias (Chang et al. 2017). All our methods perform better than the baselines. AnnealPDS performs best. For better visualization, Panel (b) shows the mean and standard deviation of our proposed Anneal PDS comparing with two baselines closely.
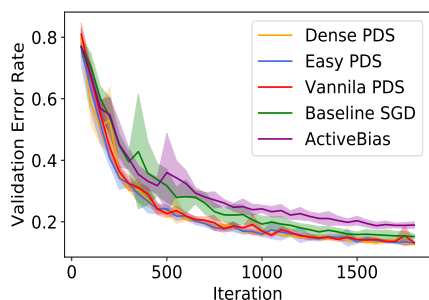

Figure 6: Speech experiment (10 repetitions). We compare the performance over the validation set of different variations of PDS with traditional SGD for every 50 iterations. Means and standard deviations are shown.

performance with fewer iterations. With a proper annealing schedule to resemble self-paced learning, we can obtain even more improvement in the final performance.

Speech Command Recognition In this section, we evaluate our method on a speech command classification task as described in (Sainath and Parada 2015). The classification task includes twelve classes: ten isolated command words, silence, or unknown class.

The database consists of 64,727 one-second-long audio recordings. As in Sainath and Parada (2015), for each recording, 40 MFCC features (Davis and Mermelstein 1980) are extracted at 10 msec time intervals resulting in 40×98 features. We use the TensorFlow implementation[4] with standard settings (see appendix). Differently from the MNIST dataset, word classes are not clearly separated and data with different mingling index values are well distributed (see appendix).

Figure 6 shows the accuracy on the validation set evaluated every 50 training iterations. Using Vanilla PDS, the model converges with fewer iterations compared to the traditional

---

[4]https://www.tensorflow.org/tutorials/audio_recognition

random sampling of mini-batches. Easy PDS and Dense PDS show similar improvement since few data have 0 mingling indices in this dataset.

As compared to the MNIST experiment, the gain of Vanilla PDS and Easy PDS is larger in this case since the dataset is more challenging. On the other hand, encouraging more difficult samples has a stronger impact on the MNIST dataset than in the Speech experiment. In all different settings, our mini-batch sampling methods are beneficial for both fast convergence and final model performance.

Discussion

In this work, we propose the use of repulsive point processes for active mini-batch sampling. We provide both theoretical and experimental evidence that using repulsive stochastic point processes can reduce the variance of stochastic gradient estimates, which leads to faster convergence. Additionally, our general framework also allows adaptive density and adaptive pair-wise interactions. This leads to further improvements in model performance thanks to balancing the information provided by the input samples, or enhancing the information around decision boundaries.

Our work is mainly focused on similarity measures in input space, which makes the algorithms efficient without additional run-time costs for learning. In future work, we will explore the use of our framework in the gradient space directly. This can potentially lead to even greater variance reduction. we believe that our proposed method may show even greater advantages in the two-stage framework such as Faster-R-CNN (Ren et al. 2015). Here, point processes can be used for region proposal sampling.

Finally, for sampling with adaptive density, we mainly use the information from mingling index, which can only be utilized for classification problems. In future work, we would also like to explore other measures such as sequences of annotations or graphs.

References

Alain, G.; Lamb, A.; Sankar, C.; Courville, A.; and Bengio, Y. 2015. Variance reduction in SGD by distributed importance sampling. *arXiv preprint arXiv:1511.06481*.

Biscio, C. A. N.; Lavancier, F.; et al. 2016. Quantifying repulsiveness of determinantal point processes. *Bernoulli*.

Bottou, L. 2010. Large-scale machine learning with stochastic gradient descent. In *COMPSTAT*. 177–186.

Buchholz, A.; Wenzel, F.; and Mandt, S. 2018. Quasi-Monte Carlo Variational Inference. In *ICML*.

Chang, H.-S.; Learned-Miller, E.; and McCallum, A. 2017. Active bias: Training a more accurate neural network by emphasizing high variance samples. *arXiv preprint arXiv:1704.07433*.

Chen, X., and Gupta, A. 2015. Webly supervised learning of convolutional networks. In *ICCV*.

Cortes, C., and Vapnik, V. 1995. Support-vector networks. *Machine learning*.

Davis, S. B., and Mermelstein, P. 1980. Comparison of parametric representations for monosyllabic word recognition in continuously spoken sentences. *IEEE Transactions on Acoustics, Speech, and Signal Processing* 28(4):357–366.

Fu, T., and Zhang, Z. 2017. CPSG-MCMC: Clustering-based preprocessing method for stochastic gradient MCMC. In *AISTATS*.

Gopal, S. 2016. Adaptive sampling for SGD by exploiting side information. In *ICML*.

Hoffman, M. D.; Blei, D. M.; Wang, C.; and Paisley, J. 2013. Stochastic variational inference. *JMLR*.

Illian, J.; Penttinen, A.; Stoyan, H.; and Stoyan, D. 2008. *Statistical analysis and modelling of spatial point patterns*, volume 70. John Wiley & Sons.

Kathuria, T.; Deshpande, A.; and Kohli, P. 2016. Batched gaussian process bandit optimization via determinantal point processes. In *NIPS*, 4206–4214.

Kita, N., and Miyata, K. 2016. Multi-class anisotropic blue noise sampling for discrete element pattern generation. *The Visual Computer*.

Krizhevsky, A.; Sutskever, I.; and Hinton, G. E. 2012. Imagenet classification with deep convolutional neural networks. In *NIPS*.

Kulesza, A.; Taskar, B.; et al. 2012. Determinantal point processes for machine learning. *Foundations and Trends® in Machine Learning* 5(2–3):123–286.

Lagae, A., and Dutré, P. 2008. A comparison of methods for generating poisson disk distributions. In *Computer Graphics Forum*, volume 27, 114–129. Wiley Online Library.

Lavancier, F.; Møller, J.; and Rubak, E. H. 2012. Statistical aspects of determinantal point processes. Technical report, Department of Mathematical Sciences, Aalborg University.

Lavancier, F.; Møller, J.; and Rubak, E. 2015. Determinantal point process models and statistical inference. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*.

LeCun, Y.; Bottou, L.; Bengio, Y.; and Haffner, P. 1998a. Gradient-based learning applied to document recognition. *Proceedings of the IEEE* 86(11):2278–2324.

LeCun, Y.; Bottou, L.; Orr, G. B.; and Müller, K.-R. 1998b. Efficient backprop. In *Neural networks: Tricks of the trade*. Springer. 9–50.

Li, H.; Wei, L.-Y.; Sander, P. V.; and Fu, C.-W. 2010. Anisotropic blue noise sampling. In *ACM TOG*.

Li, C.; Jegelka, S.; and Sra, S. 2015. Efficient sampling for k-determinantal point processes. *arXiv preprint arXiv:1509.01618*.

Macchi, O. 1975. The coincidence approach to stochastic point processes. *Advances in Applied Probability*.

Mikolov, T.; Chen, K.; Corrado, G.; and Dean, J. 2013. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*.

Møller, J., and Waagepetersen, R. P. 2004. *Statistical inference and simulation for spatial point processes*. Chapman & Hall/CRC, 2003.

Öztireli, A. C., and Gross, M. 2012. Analysis and synthesis of point distributions based on pair correlation. *ACM Trans. Graph.*

Ren, S.; He, K.; Girshick, R.; and Sun, J. 2015. Faster R-CNN: Towards real-time object detection with region proposal networks. In *NIPS*.

Ripley, B. D. 1976. The second-order analysis of stationary point processes. *Journal of applied probability*.

Roveri, R.; Öztireli, A. C.; and Gross, M. 2017. General point sampling with adaptive density and correlations. *Eurographics*.

Sainath, T. N., and Parada, C. 2015. Convolutional neural networks for small-footprint keyword spotting. In *Interspeech*.

Shrivastava, A.; Gupta, A.; and Girshick, R. 2016. Training region-based object detectors with online hard example mining. In *CVPR*.

Yin, D.; Pananjady, A.; Lam, M.; Papailiopoulos, D.; Ramchandran, K.; and Bartlett, P. 2017. Gradient diversity: a key ingredient for scalable distributed learning. In *NIPS WS*.

Zhang, C.; Bütepage, J.; Kjellström, H.; and Mandt, S. 2017a. Advances in variational inference. *arXiv preprint arXiv:1711.05597*.

Zhang, C.; Kjellström, H.; and Mandt, S. 2017b. Determinantal point processes for mini-batch diversification. In *UAI*.

Zhao, P., and Zhang, T. 2014. Accelerating mini-batch stochastic gradient descent using stratified sampling. *arXiv:1405.3080*.