Adversarial Distillation of Bayesian Neural Network Posteriors

Kuan-Chieh Wang[†‡], Paul Vicol[†‡], James Lucas[†‡], Li Gu[†], Roger Grosse[†‡], Richard Zemel[†‡]

[†]University of Toronto [‡]Vector Institute

Motivation

- Bayesian neural networks (BNNs) are a principled way to reason about uncertainty.
- MCMC methods allow us to sample from the posterior, but have high storage cost.

Summary

- We introduce a framework called Adversarial Posterior Distillation (APD) that uses a Generative Adversarial Network (GAN) to model the BNN posterior.
- We show that APD performs as well as the original posterior samples in the following standard testbeds for BNNs while using less storage:
  - Anomaly detection
  - Active Learning (exploration)
  - Defense against adversarial attacks
- We analyze the suitability of using GANs for APD.

Background

- Stochastic Gradient Langevin Dynamics (SGLD) is an MCMC method that works with mini-batches:

$$\Delta\theta^t = \frac{\epsilon^t}{2}\left(\nabla \log p(\theta^t) + \frac{N}{n}\sum_{i=1}^{n} \nabla \log p(y_i^t|x_i^t, \theta^t)\right) + \eta^t$$

- GANs can sample from rich posterior distributions. We used the WGAN with gradient penalty (WGAN-GP).

Method

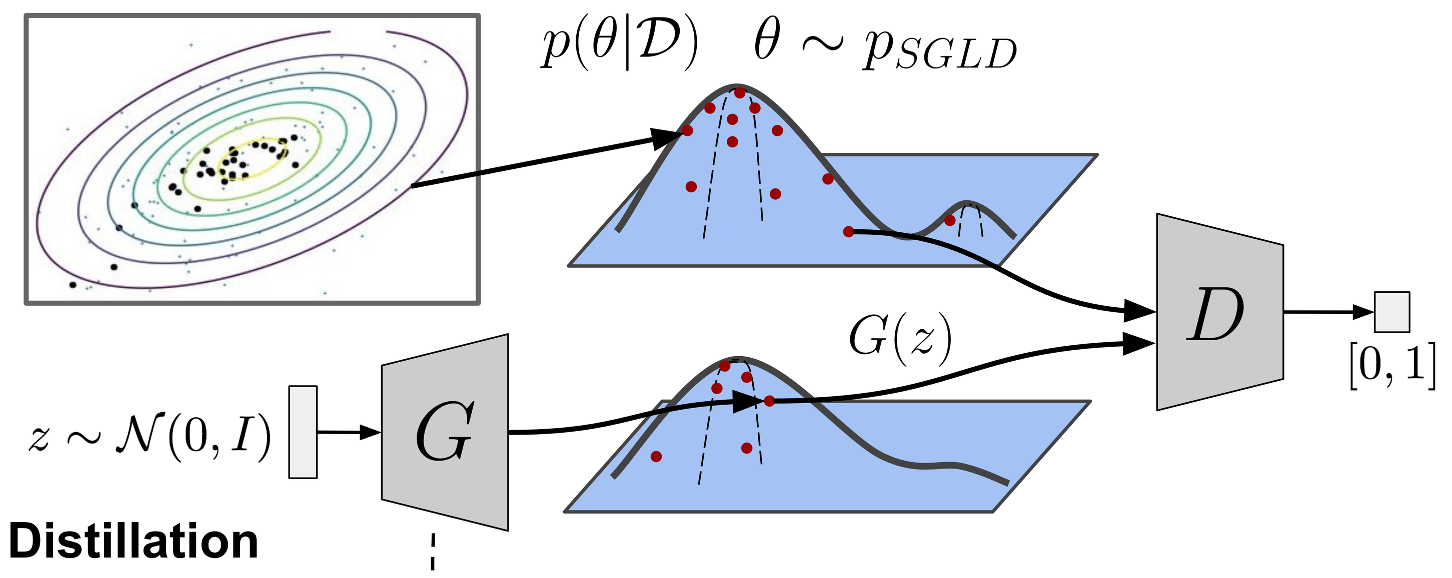
Algorithm Offline APD Distillation

1: Sample $\{\theta^t\}_{t=1}^{T}$ using MCMC updates, where T denotes the number of updates.

2: Optimize G with WGAN-GP loss using $\{\theta^t\}_{t=1}^{T}$ as real data.

- Online algorithm has sampling and GAN updates interleaved

Method (Cont.)



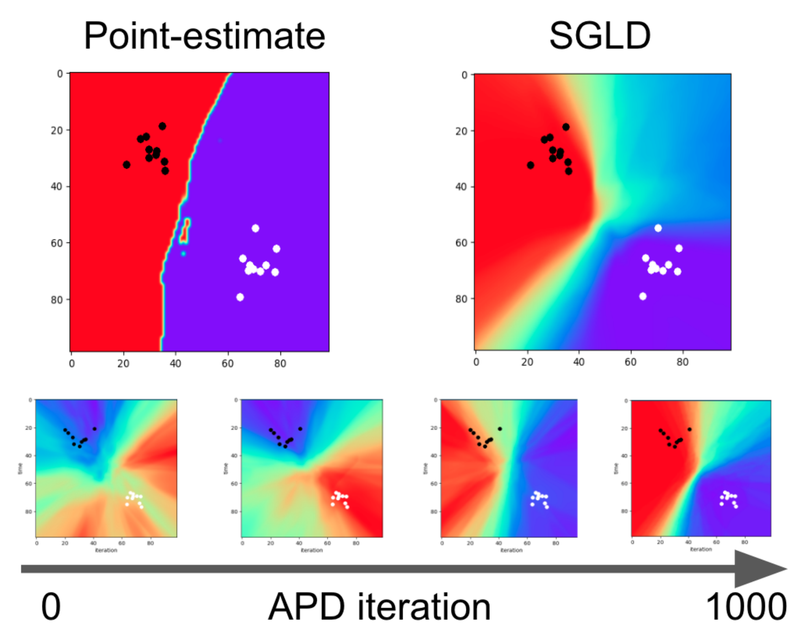
Inference

SGLD $\quad p(y|x, \mathcal{D}) \approx \frac{1}{T}\sum_t p(y|x, \theta^t), \theta^t \sim p(\theta|\mathcal{D})$

APD $\quad p(y|x, \mathcal{D}) \approx \frac{1}{T}\sum_t p(y|x, G(z^t)), z^t \sim \mathcal{N}(0, I)$

Toy Example

- Problem Setup: Classify mixture of 2 Gaussians
- The deterministic network has a hard decision boundary, while SGLD is uncertain away from data.
- APD gradually learned to model SGLD.



Anomaly Detection

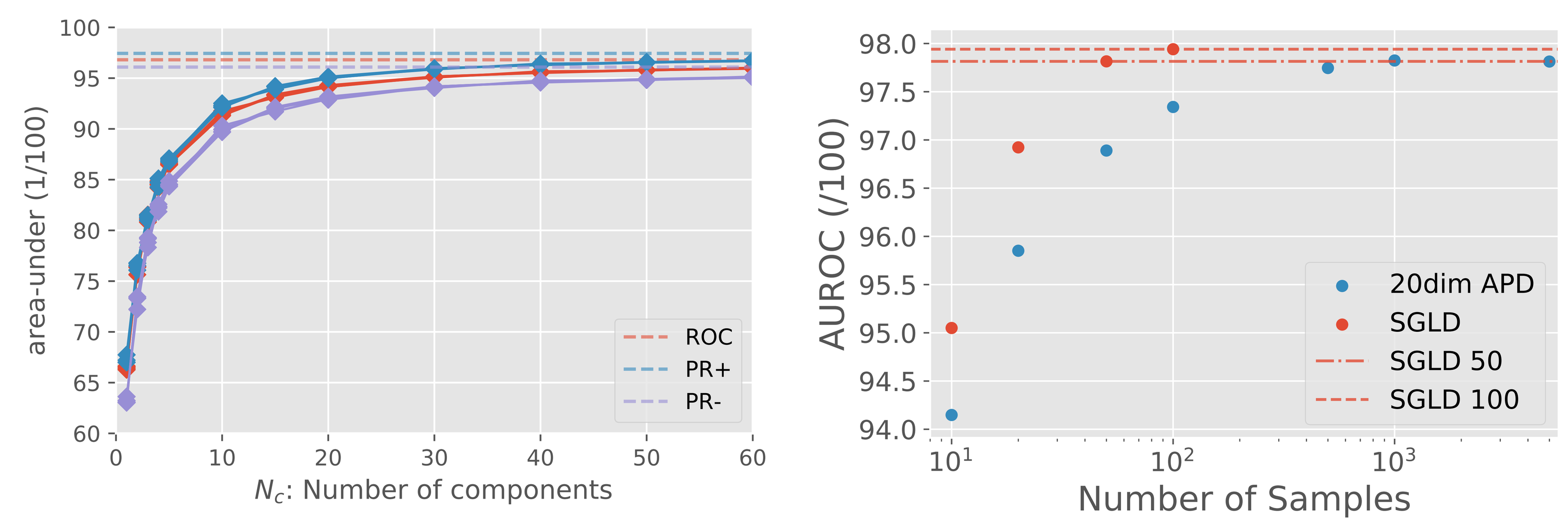
- Task: train only on in-distribution data (i.e. MNIST), and evaluate detection of out-of-distribution data.
- Model: fully connected neural network (784-400-400-10)

	Dataset	SGD			MC-Dropout			SGLD			APD (Ours)		
Det.	**area under**	**ROC**	**PR+**	**PR-**	**ROC**	**PR+**	**PR-**	**ROC**	**PR+**	**PR-**	**ROC**	**PR+**	**PR-**
	notMNIST	64.2	67.6	54.4	88.0	87.2	82.1	98.1	97.8	98.3	97.8	97.4	98.1
	OmniGlot	84.2	84.9	78.7	91.5	90.8	90.3	99.0	98.8	99.1	98.8	98.6	99.1
VR	**CIFAR10bw**	61.4	66.1	52.2	90.1	88.5	86.5	97.4	97.0	97.5	96.9	96.5	96.7
	Gaussian	67.3	70.2	57.4	91.3	89.8	89.0	99.6	99.6	99.7	99.6	99.5	99.6
	Uniform	85.4	80.7	85.8	93.6	91.2	94.8	99.8	99.8	99.9	99.8	99.7	99.8

- VR stands for variations-ratio

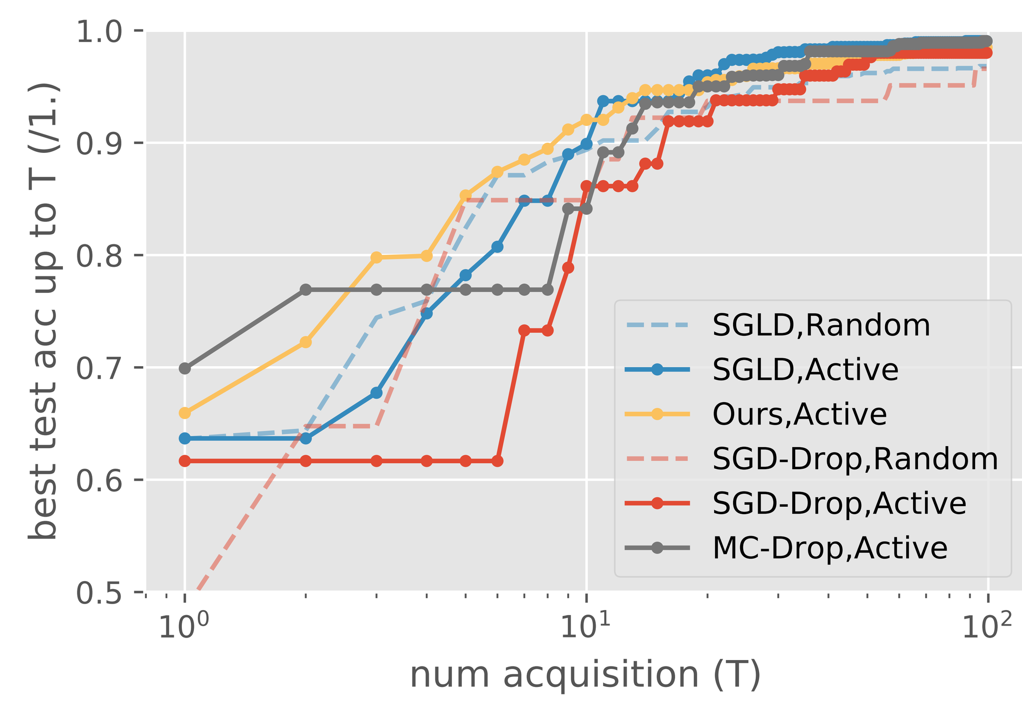
Why GANs? / Storage Savings

- Anomaly detection with increasing number of GMM components
- With APD, the storage cost (i.e., generator size) is fixed



Active Learning

- For BNNs, active learning using entropy was able to learn faster than random acquisition.



Adversarial Example Detection - MNIST

- We measured the AUROC for FGSM and PGD adversaries under each source model.

- 'Source' refers to the network used to generate attacks
- Here we used approximate model variance, $U(x)$:

Source	Attack Type	MC-Drop	SGLD	Ours
MC-Drop	**FGSM**	89.53	**94.01**	91.70
	PGD	88.37	**93.95**	91.63
SGLD	**FGSM**	54.99	**83.76**	75.93
	PGD	56.91	**84.98**	82.80
Ours	**FGSM**	54.51	83.05	**86.02**
	PGD	54.98	88.01	**93.15**

$$U(x) = \frac{1}{T}\sum_{t=1}^{T} {}^{T}_{t}\,{}_{t} - (\frac{1}{T}\sum_{t=1}^{T} {}_{t})^{T}(\frac{1}{T}\sum_{t=1}^{T} {}_{t}) \quad (1)$$