The Amortized Bootstrap

Eric Nalisnick [1] **Padhraic Smyth** [1]

Abstract

We use amortized inference in conjunction with implicit models to approximate the bootstrap distribution over model parameters. We call this the *amortized bootstrap*, as statistical strength is shared across dataset replicates through a meta-model. At test time, we can then perform *amortized bagging* by drawing multiple samples from the implicit model. We find amortized bagging outperforms bagging with independent parameter estimates across a variety of prediction tasks.

1. Introduction

Obtaining uncertainty estimates is increasingly important in modern machine learning, especially as models are given an increasing amount of responsibility. For example, obtaining uncertainty estimates for a model tasked with driving a car autonomously is essential for a safe implementation that supports interpretable logs and diagnostics. Yet, as the tasks undertaken by automation become more complex, so do the models and accompanying inference strategies. In fact, exact inference is often impossible in practice for modern models (such as deep neural networks). Thus, performing *approximate* inference well has become essential.

Recent work has enabled approximate inference methods to become increasingly efficient and expressive by using *amortized inference* and *implicit approximations* respectively. Regarding the former, amortized inference uses a shared meta-model to output posterior samples (Ranganath et al., 2016; Li et al., 2017) or the distribution of local variables (Kingma & Welling, 2014). By tying these usually independent quantities together, amortized inference is less costly but also less expressive than performing independent inferences. However, parametrizing the meta model with a neural network and optimizing with shared gradients seem to offset the decreased expressivity in practice. Turning to the latter, implicit approximations are models without explicit density functions but from which we can easily draw samples, usually by passing in a random seed to a black-box function (such as a neural network). Generative Adversarial Networks (Goodfellow et al., 2014) represent the best known implementation of an implicit model as of late, but the idea of using simulators in statistical inference is much older (Laplace, 1814; Fisher, 1930). Implicit models have been recently used within Bayesian inference for samples from a variational posterior (Ranganath et al., 2016; Tran et al., 2017; Huszár, 2017; Mescheder et al., 2017), samples from a Markov chain (Li et al., 2017), and messages passed over a factor graph (Karaletsos, 2016).

Given these successes within the Bayesian paradigm, in this paper we investigate employing amortized inference and implicit models for Frequentist inference. Specifically, we use an implicit model to approximate the bootstrap distribution over model parameters. This allows the bootstrap distribution to be captured by a parametric artifact from which we can draw an unrestricted number of samples, and yet, by using an implicit model, the approximation is free of strong distributional assumptions, thus staying true to the Frequentist paradigm. For experiments, we train an implicit bootstrap model and test its performance via bagging. We find bagging via the amortized bootstrap results in performance superior to the traditional bootstrap for models ranging from linear regression to neural networks.

2. Background

We begin by establishing notation and reviewing bootstrapping (Efron & Tibshirani, 1994) and bagging (Breiman, 1996). Let the data be denoted $\mathbf{x} \in \mathcal{X}$ with $\mathcal{X}$ representing the population. We observe N independently and identically distributed draws from $\mathcal{X}$, and these draws constitute the empirical dataset $\mathbf{X}_0 = \{\mathbf{x}_{0,1}, \dots, \mathbf{x}_{0,N}\}$. Define the data model (likelihood function) to be $p(\mathbf{x}|\boldsymbol{\theta})$ where $\boldsymbol{\theta} \in \boldsymbol{\Theta}$ are the model parameters taking values in some space $\boldsymbol{\Theta}$, and let the dataset likelihood be denoted $p(\mathbf{X}_{\cdot}|\boldsymbol{\theta}) = \prod_{i=1}^{N} p(\mathbf{x}_{\cdot,i}|\boldsymbol{\theta})$. We will also consider supervised learning tasks in which the data comes as feature-label tuples $\mathbf{D}_0 = \{(\mathbf{x}_{0,1}, y_{0,1}), \dots, (\mathbf{x}_{0,N}, y_{0,N})\}$ and the model takes the form $p(y|\mathbf{x}, \boldsymbol{\theta})$.

The Bootstrap. The Frequentist paradigm assumes there exists some $\boldsymbol{\theta}^* \in \boldsymbol{\Theta}$ that is the 'true' parameter in the sense

[1]University of California, Irvine, CA, USA. Correspondence to: Eric Nalisnick <enalisni@uci.edu>.

ICML 2017 Workshop on Implicit Models.

that it generates the population, i.e. $\mathbf{x} \sim p(\mathbf{x}|\boldsymbol{\theta}^*)\ \forall \mathbf{x} \in \mathcal{X}$. As $N \to \infty$, maximum likelihood (ML) estimation recovers $\boldsymbol{\theta}^*$. *The bootstrap* (Efron & Tibshirani, 1994) is a Frequentist procedure for handling small datasets for which we cannot rely on ML estimation to recover $\boldsymbol{\theta}^*$.

The bootstrap uses sampling from the empirical dataset to mimic sampling from the population. Doing so provides a distribution over parameters that is a proxy for the distribution created by the sampling process. Averaging over this distribution can then lead to more robust inference. To define the bootstrap formally, *bootstrap resampling* creates a new dataset $\mathbf{X}_j = \{\mathbf{x}_{j,1}, \ldots, \mathbf{x}_{j,N}\}$ by sampling with replacement N times from $\mathbf{X}_0$:

$$\mathbf{x} \sim G(\mathbf{x}) = \frac{1}{N}\sum_{i=1}^{N} \delta_{\mathbf{x}_{0,i}} \tag{1}$$

where $\delta_{\mathbf{x}_{0,i}}$ is the delta function placed at the ith example in the original dataset. The *bootstrap distribution* is then defined as

$$\boldsymbol{\theta} \sim F(\boldsymbol{\theta}) = \frac{1}{K}\sum_{k=1}^{K} \delta_{\hat{\boldsymbol{\theta}}_k} \tag{2}$$

where $\hat{\boldsymbol{\theta}}_k$ is the ML estimate for the kth (out of K) bootstrap generated dataset $\mathbf{X}_k$. Once obtained, $F(\boldsymbol{\theta})$ can be used for uncertainty quantification.

Bagging. *Bagging* (Breiman, 1996)—short for *bootstrap aggregation*—is an extension to the bootstrap for the creation of ensembles. At test time, predictions are created by averaging over $F(\boldsymbol{\theta})$—either by averaging the predictions in regression settings or by voting in classification settings.

3. Amortized Bootstrap

Our aim is to make $F(\boldsymbol{\theta})$ a parametric artifact that does not make strong distributional assumptions but is still easy to train. To this end with use an *implicit* model for $F(\boldsymbol{\theta})$, as we describe next.

Implicit Bootstrap Model. We define the *amortized bootstrap* by parametrizing the bootstrap distribution $F(\boldsymbol{\theta})$ with a differentiable function $f_{\boldsymbol{\phi}}$ with parameters $\boldsymbol{\phi}$. Samples are drawn from the bootstrap distribution via $\hat{\boldsymbol{\theta}} = f_{\boldsymbol{\phi}}(\boldsymbol{\xi})$ where $\boldsymbol{\xi} \sim p_0$ with p_0 being some fixed distribution. In this paper, we take $f_{\boldsymbol{\phi}}$ to be a neural network (NN). Thus we define the bootstrap distribution $F(\boldsymbol{\theta})$ *implicitly* through the functional sampler $f_{\boldsymbol{\phi}}$—a so-called *implicit model* (Mohamed & Lakshminarayanan, 2016). In other words, we cannot evaluate F but we can sample from it easily, similarly to the generative component of a Generative Adversarial Network (GAN) (Goodfellow et al., 2014) or the posterior defined by variational programs (Ranganath et al., 2016).

Algorithm 1 Amortized Bootstrap Learning

Input: Max iterations T, learning rate η, number of replications K, dataset $\mathbf{X}_0$, data model $p(\mathbf{x}|\boldsymbol{\theta})$, sampler $f_{\boldsymbol{\phi}}$, fixed distribution $p_0(\boldsymbol{\xi})$.
for $t = [1, T]$ **do**
  $\{\mathbf{X}_1, \ldots, \mathbf{X}_K\} \sim G(\mathbf{x})$ *(sample every few epochs)*
  $\{\boldsymbol{\xi}_1, \ldots, \boldsymbol{\xi}_K\} \sim p_0(\boldsymbol{\xi})$ *(sample for each minibatch)*
  $\boldsymbol{\theta}_{\boldsymbol{\phi},k} = f_{\boldsymbol{\phi}}(\boldsymbol{\xi}_k)\ \ \forall k \in [1, K]$
  $\boldsymbol{\phi}_t \leftarrow \boldsymbol{\phi}_{t-1} + \eta\, \partial \mathcal{J}(\mathbf{X}_0, \boldsymbol{\phi}_{t-1})/\partial \boldsymbol{\phi}_{t-1}$
end
Return amortized bootstrap parameters $\boldsymbol{\phi}_T$

Optimization Objective. The parameters $\boldsymbol{\phi}$ can be estimated by optimizing the likelihood function under samples from the implicit model:

$$\begin{aligned}\mathcal{J}(\mathbf{X}_0, \boldsymbol{\phi}) &= \mathbb{E}_{F(\boldsymbol{\theta})}\mathbb{E}_{G(\mathbf{x})}\left[\log p(\mathbf{X}|\boldsymbol{\theta})\right] \\ &\approx \frac{1}{K}\sum_{k=1}^{K} \log p(\mathbf{X}_k|\hat{\boldsymbol{\theta}}_{\boldsymbol{\phi},k})\end{aligned} \tag{3}$$

where $\mathbf{X}_k \sim G(\mathbf{x})$ and $\hat{\boldsymbol{\theta}}_{\boldsymbol{\phi},k} = f_{\boldsymbol{\phi}}(\hat{\boldsymbol{\xi}}_k)$. Notice that this objective is the same as that of the traditional bootstrap: if the K models are independent, maximizing the sum is equivalent to maximizing each term individually.

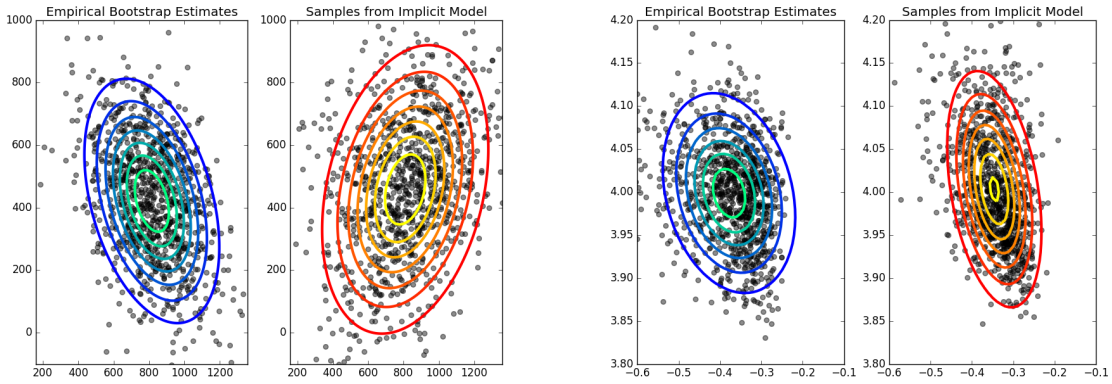
Gradient-Based Optimization. Learning the amortized bootstrap distribution amounts to maximizing Equation 3 with respect to $\boldsymbol{\phi}$. We assume the model parameters $\boldsymbol{\theta}$ are continuous and thus can take gradients directly through the parameter samples and then into $f_{\boldsymbol{\phi}}$ as follows:

$$\frac{\partial \mathcal{J}(\mathbf{X}_0, \boldsymbol{\phi})}{\partial \boldsymbol{\phi}} = \frac{1}{K}\sum_{k=1}^{K} \frac{\partial \log p(\mathbf{X}_k|\hat{\boldsymbol{\theta}}_{\boldsymbol{\phi},k})}{\partial \hat{\boldsymbol{\theta}}_{\boldsymbol{\phi},k}} \frac{\partial \hat{\boldsymbol{\theta}}_{\boldsymbol{\phi},k}}{\partial \boldsymbol{\phi}}. \tag{4}$$

The optimization procedure is summarized in Algorithm 1. The user must specify how often to sample the data and parameters from $G(\mathbf{x})$ and $F(\boldsymbol{\theta})$ respectively. We found that sampling new parameters for every minibatch and sampling new datasets after every few epochs (3 to 5) works well.

Amortized Bagging. Using the amortized bootstrap for *amortized bagging* is straightforward. At test time, we draw some number of samples M from $f_{\boldsymbol{\phi}}$—$\{\hat{\boldsymbol{\theta}}_1, \ldots, \hat{\boldsymbol{\theta}}_M\} = f_{\boldsymbol{\phi}}(\boldsymbol{\xi}_m)$—and then we ensemble the M models $p(\mathbf{X}|\hat{\boldsymbol{\theta}}_m)$ via the usual mechanisms (voting, averaging, etc.). Here we explicitly see the benefits of using the amortized bootstrap: the size of the ensemble M can be larger than the number of replications used during training. This is not possible with traditional bagging.

Theoretical Underpinnings. A natural question to ask is: does the underlying theory for the bootstrap still hold in the amortized version? In short, the answer is 'no', as the estimators are no longer independent, instead linked through $\boldsymbol{\phi}$. In this paper, our aim to show that the amortized boot-



(a) Diabetes. (b) Boston Housing.

Figure 1. Distribution Over 2D Linear Models. 1000 samples from the empirical bootstrap distribution (blue contours) vs. the amortized bootstrap distribution (red contours) are shown for the diabetes (a) and housing (b) datasets reduced to two dimensions. Distributions are similar but not identical.

strap produces an efficient ensembling strategy that performs well on out-of-sample prediction tasks. We leave to future work assessing the amortized bootstrap's ability to quantify uncertainty.

4. Experiments

We next perform experimental evaluation of the amortized bootstrap, concentrating on bagging performance. We ensembled models by averaging predictions in regression experiments and by voting in classification tasks. Regarding hyperparameter selection, AdaM (Kingma & Ba, 2014) was used for all experiments with a cross-validated learning rate (other parameters were kept at Tensorflow defaults). We used $K = 3$ (in Equation 3) and resampled the data every three epochs when training all amortized bootstrap models. All implicit models were chosen to be one-hidden-layer neural networks with a standard Gaussian input distribution.

4.1. Diagnostics

We first perform some diagnostics, checking whether the method learns a distribution similar to the true bootstrap distribution and if the distribution's variance decreases as the number of data points grows.

Visual Sanity Check. We begin analysis with a small-scale linear regression experiment on the diabetes (Efron et al., 2004) and Boston housing (Lichman, 2013) datasets. We reduce the input features to only, in the case of the former, the body mass index and blood pressure attributes and, in the case of the latter, the crime rate and number of rooms. Reducing the features to two dimensions allows the parameter estimates to be visualized in order to compare the traditional and amortized bootstrap distributions. For each dataset, we generated 1000 replicate datasets via bootstrap resampling, keeping their size the same as the original training set's, and calculated the OLS solution for each. The implicit model used a 25 dimensional $\boldsymbol{\xi}$ and contained 50 hidden units. We drew 1000 samples from the implicit model to compare them with the 1000 estimates computed via the traditional bootstrap.

Figure 1 shows the results. Subfigure (a) contains a scatter plot of the parameter estimates (empirical on left, implicit on right) computed on the diabetes dataset, and subfigure (b) shows the same quantities on the housing dataset. Gaussian distributions are fit to each set of samples to aid visual comparison. We see that the locations and scales of the distributions are nearly identical, but the covariance structures differ. For diabetes, one tilts to the left while the other to the right, and for housing, the amortized distribution is more narrow. Understanding the precise nature of these discrepancies is a topic for future work.

Varying Dataset Size. Next we check if the amortized bootstrap exhibits a key property of the traditional version: variance decreases with dataset size, as the ML estimate becomes more reliable. We use the cover-type dataset (Lichman, 2013) to analyze this property as the dimensionality is moderate (54), and there are enough instances (522,910) that the ML estimate achieves near optimal performance. We used an implicit model with 500 dimensional $\boldsymbol{\xi}$ and 1000 hidden units. Figure 2 shows the results, plotting the trace of each distribution's empirical covariance matrix as dataset size increases. We see that the amortized bootstrap has a smaller variance for smaller datasets but still exhibits the decrease as expected, nearly matching the traditional bootstrap for $N \geq 1000$.

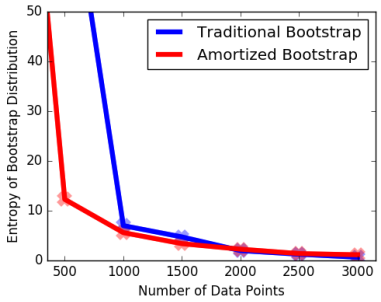
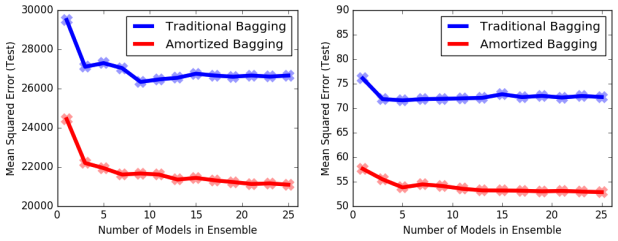


Figure 2. Variance of Bootstrap Distribution. The plot above shows the trace of the empirical covariance matrix (i.e. trace($\hat{\Sigma}$) of 1000 samples as training set size increases from 500 to 3000 instances. We see, for both types of bootstrapping, variance decreases as expected.



(a) Diabetes. (b) Boston Housing.

Figure 3. Bagged Linear Models for 2D Datasets. Mean squared error on test set (y-axis) for various ensemble sizes (x-axis) is shown for 2D diabetes and housing datasets. Amortized is superior for all K.

4.2. Bagging Performance

In the final set of experiments, we test bagging performance under the traditional and amortized bootstraps. We use the diabetes, Boston housing, and cover-type datasets mentioned previously and add rotated MNIST (Larochelle et al., 2007).

Linear Models. We first examine bagging performance on the 2D regression datasets studied in the diagnostics. Mean squared error (MSE) on the test set for an increasing ensemble size is shown in Figure 3 (diabetes in (a), housing in (b)). Amortized bagging outperforms traditional bagging for all ensemble sizes and shows decreasing error as the ensemble grows. Thus, while the amortized bootstrap does not recover the true bootstrap distribution, the distribution it does find has better predictive accuracy.

Logistic Regression. Next we examine logistic regression models trained on cover-type. We plot test set classification error not only as ensemble size (K) increases but for the various dataset sizes (N) examined in the variance diagnostics. Figure 4 shows the results for $N \in [500, 1500, 2500]$.

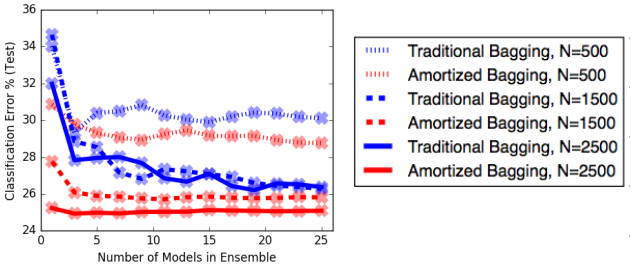


Figure 4. Bagged Logistic Regression for Cover Type Dataset. Classification error on test set (y-axis) for various ensemble sizes (x-axis) is shown for cover type dataset reduced to 500, 1500, and 2500 training instances.

	Test Error for Ensemble of Size K		
	$K = 1$	$K = 5$	$K = 25$
Bagged NNs, Traditional	22.57	19.68	18.57
Bagged NNs, **Amortized**	17.03	16.82	**16.18**

Table 1. Bagged Neural Networks for Rotated MNIST. Classification error (percentage) on test set for various ensemble sizes (K) is shown for the rotated MNIST dataset ($N_{\text{train}} = 12,000$, $N_{\text{test}} = 50,000$). Amortized is superior for all K.

We see that amortized bagging outperforms traditional bagging for all dataset and ensemble sizes except at ($K = 3$, $N = 500$). Moreover, amortized bagging with 1500 data points has lower error for all K than traditional bagging using 2500 data points.

Neural Networks. Lastly, we test bagging performance for neural networks with one hidden layer of 2000 units and no regularization. We trained on the rotated MNIST dataset, which presents a challenge for a vanilla feedforward network (especially without regularization) as there are only 12,000 training images. We used an implicit model with 50 dimensional $\boldsymbol{\xi}$ and 10 hidden units (due to memory constraints). Test set classification error for ensembles of size 1, 5, and 10 are shown in Table 1. We see that, again, amortized bagging outperforms the traditional bootstrap at all ensemble sizes.

5. Conclusions

We have described how to use amortized inference and implicit models to approximate the bootstrap distribution. We empirically demonstrated the method is effective for bagging, achieving superior test set performance across all model types and ensemble sizes we tested. In future work, we plan to better assess the amortized bootstrap's ability to quantify uncertainty as well as to expand the bagging results to larger models and more challenging datasets.

Acknowledgements

We thank DeepMind for the travel award.

References

Breiman, Leo. Bagging predictors. *Machine learning*, 24 (2):123–140, 1996.

Efron, Bradley and Tibshirani, Robert J. *An introduction to the bootstrap*. CRC press, 1994.

Efron, Bradley, Hastie, Trevor, Johnstone, Iain, and Tibshirani, Robert. Least angle regression. *The Annals of statistics*, 32(2):407–499, 2004.

Fisher, Ronald A. Inverse probability. *Mathematical Proceedings of the Cambridge Philosophical Society*, 26: 528–535, 1930.

Goodfellow, Ian, Pouget-Abadie, Jean, Mirza, Mehdi, Xu, Bing, Warde-Farley, David, Ozair, Sherjil, Courville, Aaron, and Bengio, Yoshua. Generative adversarial nets. *Neural Information Processing Systems (NIPS)*, 2014.

Huszár, Ferenc. Variational inference using implicit distributions. *ArXiv Preprint*, 2017.

Karaletsos, Theofanis. Adversarial Message Passing For Graphical Models. *NIPS 2016 Workshop on Advances in Approximate Bayesian Inference*, 2016.

Kingma, Diederik and Ba, Jimmy. Adam: A method for stochastic optimization. *International Conference on Learning Representations (ICLR)*, 2014.

Kingma, Diederik and Welling, Max. Auto-encoding variational Bayes. *International Conference on Learning Representations (ICLR)*, 2014.

Laplace, Pierre-Simon. A philosophical essay on probabilities. 1814.

Larochelle, Hugo, Erhan, Dumitru, Courville, Aaron, Bergstra, James, and Bengio, Yoshua. An empirical evaluation of deep architectures on problems with many factors of variation. In *Proceedings of the 24th international conference on Machine learning*, pp. 473–480, 2007.

Li, Yingzhen, Turner, Richard E, and Liu, Qiang. Approximate inference with amortised mcmc. *ArXiv Preprint*, 2017.

Lichman, M. UCI machine learning repository, 2013. URL http://archive.ics.uci.edu/ml.

Mescheder, Lars M., Nowozin, Sebastian, and Geiger, Andreas. Adversarial variational bayes: Unifying variational autoencoders and generative adversarial networks. *ArXiv Preprint*, 2017.

Mohamed, Shakir and Lakshminarayanan, Balaji. Learning in implicit generative models. *ArXiv Preprint*, 2016.

Ranganath, Rajesh, Tran, Dustin, Altosaar, Jaan, and Blei, David. Operator variational inference. In *Advances in Neural Information Processing Systems*, pp. 496–504, 2016.

Tran, Dustin, Ranganath, Rajesh, and Blei, David M. Deep and hierarchical implicit models. *ArXiv Preprint*, 2017.