Large-Scale Supervised Learning of the Grasp Robustness of Surface Patch Pairs

Daniel Seita[1], Florian T. Pokorny[1,3], Jeffrey Mahler[1], Danica Kragic[3], Michael Franklin[1], John Canny[1], Ken Goldberg[1,2]

***Abstract*— The robustness of a parallel-jaw grasp can be estimated by Monte Carlo sampling of perturbations in pose and friction but this is not computationally efficient. As an alternative, we consider fast methods using large-scale supervised learning, where the input is a description of a local surface patch at each of two contact points. We train and test with disjoint subsets of a corpus of 1.66 million grasps where robustness is estimated by Monte Carlo sampling using Dex-Net 1.0. We use the BIDMach machine learning toolkit to compare the performance of two supervised learning methods: Random Forests and Deep Learning. We find that both of these methods learn to estimate grasp robustness fairly reliably in terms of Mean Absolute Error (MAE) and ROC Area Under Curve (AUC) on a held-out test set. Speedups over Monte Carlo sampling are approximately 7500x for Random Forests and 1500x for Deep Learning.**

I. Introduction

In many applications, such as warehouse order fulfillment and home decluttering, robots must be able to reliably grasp a variety of objects in the presence of uncertainty in pose and friction. Many robotic grasp synthesis algorithms are based on grasp quality metrics derived from exact force closure analysis [1], [2]. Recent work has formulated noise-models on grasp parameters [3], [4] by treating the probability of force closure as a random variable and sampling to estimate the robustness of a grasp.

Random Forests are known to be effective for many Big Data classification tasks [5], [6]. Deep Learning with large datasets has provided impressive results in image classification [7], [8], speech recognition [9], [10], and in some reinforcement learning contexts [11], [12]. In this work, we explore how Big Data machine learning can enable these large scale supervised learning methods, Random Forests and Deep Learning, to rapidly estimate grasp robustness. Fast and accurate grasp robustness evaluation would be helpful for real-time grasp adaptation algorithms [13] and would also be useful components of robot grasp planners taking point clouds as input.

[1]The AUTOLAB (automation.berkeley.edu), Department of Industrial Engineering and Operations Research and Electrical Engineering and Computer Sciences; {seita, jmahler, franklin, canny, goldberg}@berkeley.edu

[2]Department of Industrial Engineering and Operations Research and Department of Electrical Engineering and Computer Sciences;

[1−2] University of California, Berkeley; Berkeley, CA 94720, USA

[3]Computer Vision and Active Perception Lab, Centre for Autonomous Systems, School of Computer Science and Communication, KTH Royal Institute of Technology, Stockholm, Sweden; {fpokorny,dani}@kth.se

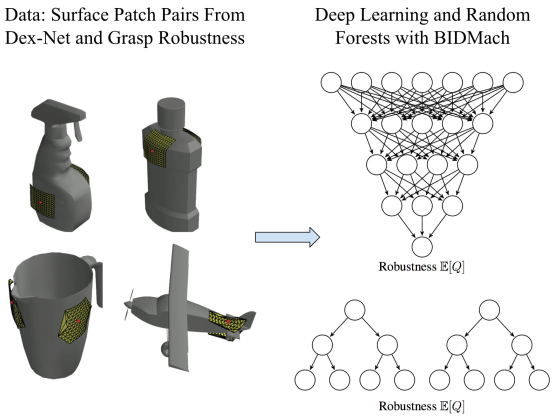

Fig. 1: Left: Training data for grasp robustness estimation is generated by Dex-Net using Monte Carlo sampling. Right: Deep Learning and Random Forests (with BIDMach) estimate grasp robustness.

Fig. 1 illustrates the approach. Let $\mathcal{X}$ parameterize a grasp in terms of a pair of local contact patches, patch centers, and normals. We consider a grasp robustness measure $Q : \mathcal{X} \rightarrow [0, 1]$ which encodes the probability of the grasp achieving force closure under random perturbations. Uncertainty is modeled probabilistically using a graphical model which takes into account uncertainty in friction, grasp, and object pose. We use a dataset of 1.66 million grasps, split into disjoint training and testing subsets, and local shape patches extracted from Dexterity Network, Dex-Net [14], a large scale grasp robustness estimator. For each grasp g, Dex-Net computes $Y = \mathbb{E}[Q(g)]$ by sampling 100 perturbations; these Y values serve as the training labels. We learn a function which maps a local grasp contact patch pair into an estimate of Q using the high-performance machine learning toolkit, BIDMach [15]. We evaluate the performance of the two methods based on Mean Absolute Error (MAE) and ROC Area Under Curve (AUC).

II. Related Work

A. Grasp Quality Assessment

Robot grasp planning analyzes the mechanical force-closure properties assuming the exact pose is known [2], [16], assuming that the object is immobilized without slip under quasistatic Coulomb friction. In this paper we explicitly consider perturbations in pose and friction to estimate the quality metric of probability of force closure.

B. Grasp Planning under Uncertainty

The computation of grasp quality measures depends on precise knowledge of quantities such as contact positions, the object's center of mass, friction coefficients, and surface normal vectors at the contact points. However, uncertainty and noise, both due to imprecision in the actuation of a robot and due to imperfect and incomplete perception of the environment by the robot's sensors is unavoidable.

One option is to use probabilistic models to determine grasps that are robust under perturbations [17], [18]. Recent work has focused on uncertainty in pose [19], [20], [21] and contact position [22]. In [3], [4], a probabilistic shape model based on Gaussian Process Implicit Surfaces was considered and [4] treated the grasp synthesis problem with a Multi-Armed Bandit model. By sampling a large number of grasps under a probabilistic model, these works considered the *probability* of force closure.

C. Big-Data and Robotic Manipulation

Scaling effects in robotic manipulation have recently received increased interest. Miller et al. [23] introduced one of the earliest grasp simulators, GraspIT, and considered large scale grasp simulation. Pokorny et al. introduced the notion of Grasp Moduli Spaces [24] jointly modeling continuous grasp and object deformations, allowing for continuous grasp optimization in this space. In experiments with 100 million grasp candidates sampled uniformly and independently from real-world shapes, the properties of the Ferrari-Canny grasp quality metric were found to vary smoothly over this space. In [25], Detry et al. studied the problem of learning a dictionary of object parts for the purpose of grasp synthesis. However these methods did not consider robustness to uncertainty.

Kappler et al. [26] studied the effectiveness of physics simulation and classical grasp quality measures on a database of 700 meshes and 300000 grasps. Using features derived from global multi-view point-cloud templates extracted along the robot hand's approach direction, the authors applied Logistic Regression and Deep Learning (using Convolutional Neural Networks) for the binary classification task of determining grasp stability. They found through crowdsourcing that simulation-based grasp stability labeling was more consistent with human labeling than evaluation using classical grasp measures. Our approach is similar except that we use a simpler featurization that primarily uses critical information about the local contact area of the grasp, which may be missing from their feature template. In addition, our dataset is more than five times larger.

Previous work explored scaling effects where large datasets of 3D object models [27], [28], [29], [30] are used to train grasping and manipulation policies for robots. In particular, Mahler et al., created Dex-Net [14], a large scale grasp and object database of more than 10000 objects and 2.5 million grasps, and showed that scaling effects in data size beyond 1000 objects led to significant improvements in a correlated bandit approach to optimal grasp selection. This work uses a *Cloud Robotics* [31] approach to manipulation, utilizing up to 1500 cloud-based compute nodes to compute optimal grasps, enabled by on-demand compute capabilities such as the Google Cloud Compute Engine and Amazon EC2. In a Cloud Robotics context, the learned model could be updated daily based on new data collected from robots in the field (reported success/failure of executed grasps) and then the model can be recomputed and downloaded by the robots to apply locally for future grasps.

D. Implementation in BIDMach

We utilize BIDMach [15], [32], a GPU-accelerated toolkit for running machine learning methods at scale, including Deep Learning and Random Forests (RFs). BIDMach processes data in minibatches for both Deep Learning and classical machine learning algorithms, including Random Forests. It can therefore process very large datasets on machines with modest memory. The minibatch design is common in Deep Learning toolkits, but not so for Random Forests. All the alternative RF implementations we are aware of require training data to fit in memory. BIDMach by contrast can process datasets up to available disk storage. The Deep Learning and Random Forest implementations are both GPU-accelerated in BIDMach, with the potential to provide up to 10x speedup over similar cost/power CPUs.

III. Definitions and Problem Statement

Our goal is to learn a function that can quickly and reliably estimate the robustness of a parallel jaw grasp, where robustness is based on the probability of force closure based on sampling perturbations in pose using features in the neighborhood of the nominal grasp contact points.

Point contact model: A point-contact parallel jaw grasp for the purpose of force-closure analysis is determined by its contact points[1] $c_1, c_2 \in \mathbb{R}^3$, associated unit inward pointing surface normals n_1, n_2 at those contacts and a friction coefficient $\mu > 0$, which we assume to be identical at both contact points. In summary, a parallel jaw grasp g is given by a tuple $g = (c_1, c_2, n_1, n_2, \mu)$ and we define the force closure indicator $\mathbb{I}_{FC}(g)$ to be a binary variable equal to 1 if g is in force closure and 0 otherwise. We emphasize that for a fixed g, $\mathbb{I}_{FC}(g)$ is deterministic.

Grasp patch pairs: Given a triangular mesh M describing an object, we consider a featurization of a local grasp patch pair characterized by its contact points $c_1, c_2 \in M$. First, we compute the surface normals n_1, n_2 at these two points. Then, we choose an orientation of the tangent plane centered at the respective points c_1, c_2 and compute $k \times k$ depth maps M_1, M_2. These maps describe the depth offset of points on the surface mesh. To reduce the dimension of the featurization, we then extract the discs D_1, D_2 inscribed within M_1, M_2. Since we consider these contacts as arising from a parallel jaw gripper, we call the tuple of features $\ell_{pp} = (g, D_1, D_2) = (c_1, c_2, n_1, n_2, \mu, D_1, D_2)$ a *local patch pair*.

[1]We focus on the case of two contact points which define the principal grasp axis for the most common robot "parallel-jaw" gripper. The principal grasp axis can also be used for multifingered robot hands, defining the pose of two fingers to constrain the placement of subsequent fingers.

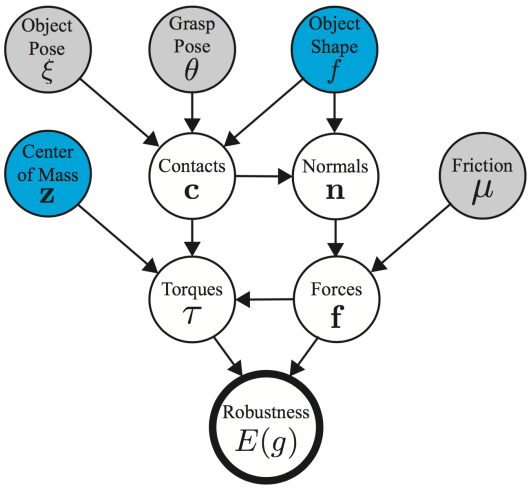

Fig. 2: Illustration of our probabilistic model representing the uncertainty in grasp quality. The shaded nodes denote observed values, and blue nodes are deterministic, known quantities. To estimate robustness, DexNet 1.0 assumes object shape is known but this will be relaxed in future work. This model is sampled 100 times for each grasp to compute $E(g)$.

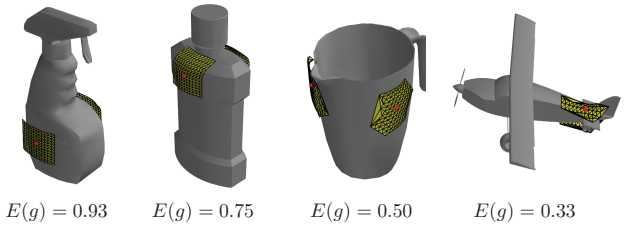

Fig. 3: Four examples of local shape patches (highlighted in yellow) on object meshes from Dex-Net, with varying $E(g)$ values. The robot's grasp makes contact with these objects at the center of their patches. Intuitively, the surface patches corresponding to more robust grasps are flatter and closer to the object's center of mass.

Probabilistic perturbation model: We model noise in perception and actuation of the robotic grippers using the probabilistic graphical model depicted in Fig. 2, which is similar to the approach in Dex-Net 1.0 [14]. A parallel jaw gripper is abstracted as a grasp line v and contact points c_1, c_2 along v. We then model noise in the grasp by means of Gaussian noise on $\mathbb{R}^6$ that is mapped to $SE(3)$, the special Euclidean group via $\exp : \mathbb{R}^6 \to SE(3)$ [33]. Similarly, noise in object orientation and position is modeled via Gaussian noise on the tangent space $\mathbb{R}^6$ of $SE(3)$. Finally, we model noise in friction coefficients μ by a Gaussian.

Given a grasp target configuration g, we denote by $G(g)$ the resulting *random variable* of grasp perturbations of g with respect to our graphical model. We denote by $E(g)$ the empirical sample estimate for the expectation of $\mathbb{I}_{FC}(G(g))$ for 100 samples that we draw. Fig. 3 illustrates exemplar patch pairs from our data, along with their $E(g)$ values.

Regression formalization: The force closure indicator $\mathbb{I}_{FC}$ under this probabilistic perturbation model is a random variable, and we explore if the quantity $\mathbb{E}[\mathbb{I}_{FC}(G(g))]$ can be determined with high precision from grasp-patch data $\ell_{pp} = (g, D_1, D_2)$ by large-scale regression. Thus, we treat

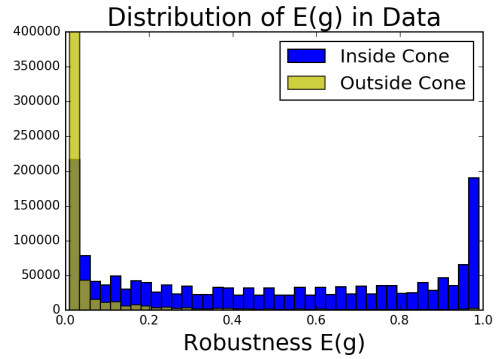

Fig. 4: Distribution of probability of force closure estimates $E(g)$ in our data, depicted as a histogram of counts. Yellow and blue indicate, respectively, grasps that do *not*, and grasps that *do*, satisfy the friction cone condition. The yellow distribution is extraordinarily skewed, with roughly two million grasps accounted for by the first histogram bin. The height is cut off at 400,000 for readability.

$\mathbb{E}[\mathbb{I}_{FC}(G(g))]$ as a deterministic function $\hat{E}$ of local patch pair information, where

$$\hat{E}(\ell_{pp}) \simeq \mathbb{E}[\mathbb{I}_{FC}(G(g))]. \quad \text{(III.1)}$$

Note that we do not know $\mathbb{E}[\mathbb{I}_{FC}(G(g))]$ exactly, and must approximate it with $E(g)$ using 100 samples. During computation of $E(g)$, our probabilistic model can occasionally generate samples lying outside of the grasp patches, since Gaussian noise has unbounded support. As an example, our model could generate contact points that lie completely outside of the patches that we use as features, resulting in an incorrect measurement of $E(g)$. However, if the grasp patch size is chosen so that sufficiently many contact pair points lie on the parameterized patches, ℓ_{pp} is expected to contain sufficient information to approximate $E(g)$.

Eliminating Non-Robust Grasps: One way to reject non-robust parallel jaw grasp candidates is to consider the following *friction cone condition*: is the line connecting the two grasp contact points within both friction cones centered at the contact points? If this is not the case for a particular grasp, then that grasp fails this condition and in the deterministic (but not probabilistic) setting, it is not in force closure. We investigate the probability of force closure distribution under this heuristic. Noisy grasps in our data that do not satisfy the friction cone condition follow the yellow distribution in Fig. 4 while grasps that satisfy it follow the blue distribution. As expected, the yellow distribution is heavily skewed with the majority of grasps having an $E(g)$ of roughly zero.

In this paper, our main interest is in understanding the properties of grasps that are likely to be robust, so we do not want the data to consist overwhelmingly of non-robust grasps. The friction cone heuristic furthermore serves as a useful filter by ensuring that even the non-robust grasps share a property common to all robust grasps. In addition, we still obtain enough non-robust grasps for a balanced analysis since the filter by itself is not strict enough to enforce robustness. (It only keeps grasps that are in force closure based on the mean over 100 trials, but individual trials may not be robust.) We therefore perform all subsequent analysis

on the 1.66 million patch pairs from the blue distribution in Fig. 4.

Evaluation Metrics: We evaluate our algorithms using two metrics: Mean Absolute Error (MAE) and Area Under the Curve (AUC), where the latter is computed from a Receiver Operating Characteristic (ROC) curve. MAE is simple and provides intuition on the distance between our predictions and the $E(g)$ values. ROCs provide richer information and also allow analysis of Type I and Type II errors, which we discuss in Section IV-E.2. ROCs are typically applied on binary classification tasks. While our targets are real-valued in $[0, 1]$, we can apply ROCs here because the targets represent a probability (a fraction of 100 trials) of $\{0, 1\}$-valued grasp outcomes, and they can be treated as 100 discrete trials for the purpose of computing ROCs.

IV. Experiments

A. Local Shape and Grasp Sample Generation

To generate a training dataset $\mathcal{D}$, we first extract approximately 1.66 million patch pairs from 10,713 mesh models from Dex-Net 1.0. The majority of the models come from the SHREC 2014 Large-Scale shape retrieval benchmark (8,987 models) [34], while the remainder of the data is sampled uniformly from the mesh models in Dex-Net 1.0.

We extract 15×15 depth-maps M_1, M_2 that span a $5.0\text{cm} \times 5.0\text{cm}$ plane tangent to the grasp approach axis on the shape surface for each contact point following the graphical model displayed in Fig. 2. (The 15×15 resolution follows the convention of Dex-Net and is in part due to our storage limitations.) We then extract the discs inscribed within M_1, M_2, forming D_1, D_2. We also add the patch orientation $v = \frac{c_1 - c_2}{\|c_1 - c_2\|_2}$ to form an initial 289-dimensional featurization $(g, v, D_1, D_2) \in \mathbb{R}^{289}$. As stated earlier, for ground truth of $\mathbb{E}[\mathbb{I}_{FC}(G(g))]$, we utilize the sample mean[2] obtained by Monte-Carlo integration over 100 samples $G(g)$ from our generative graphical model.

We divide our data of 1.66 million grasps into a training set of about 1.55 million grasps, a validation set of exactly 29790 grasps (about 2% of training size), and a testing set of exactly 79000 data points. This is roughly a 95%-5% training/validation versus testing split, though in some experiments, we do not use the full training data so that we can observe the effect of increasing training data size on a fixed-size testing set. To avoid object overlap between training/validation and testing datasets, all patches generated from a particular Dex-Net object are entirely in the training/validation or entirely in the testing set.

B. Fixed Constant Predictor Baseline

To understand the relative benefit of our predictors, it is useful to compare MAE results with a fixed constant baseline. One way to do this is to compute the median $E(g)$ of the expected grasp quality in the entire training data, and return a constant function which always outputs that median as the prediction. This is a reasonable baseline because the MAE of a single, scalar approximation of a set of scalar-valued targets is minimized with the median of the elements. Doing this, we get the baseline performance of ≈ 0.316 MAE. In addition, the AUC of the corresponding ROC is 0.501, roughly at the level of random guessing.

[2]The estimated standard error of this mean is at most $\sqrt{0.25}/\sqrt{100} = 0.05$, where 0.25 is the largest sample variance we observed (among 100 samples) in the predicted robustness for any grasp in our data.

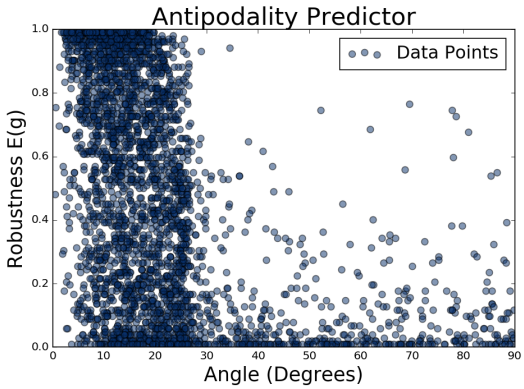

Fig. 5: Scatter plot of a random subset of our training grasps with respect to their "antipodality" angle (i.e, the angle between the grasp axis and the surface normals) and robustness. Darker areas correspond to a greater density of grasps. For the antipodality-based estimator, we use the best-fit line from the complete training data to predict robustness for testing grasps.

C. Antipodality-Based Estimator

As a first attempt at estimating $E(g)$ we measure the angle between the grasp axis and the surface normals. This is the angle of the smallest friction cone such that the grasp is in force closure; it can be thought of as a continuous metric on antipodality. Fig. 5 plots a random subset of our training data grasps with respect to this computed angle (or "antipodality") and robustness.

As can be seen, there is only a weak linear correlation between antipodality and $E(g)$: the correlation coefficient is -0.46. This result is encouraging because if the computed angle is small, the grasp should theoretically be in force closure over larger perturbations of the surface normals and grasp axis. As evident by the left portion of Fig. 5, however, there is a huge spread of grasp robustness with small angles. This suggests that our best-fit line predictor from just the antipodality feature (plus a bias term) is limited and that we need more information to better estimate $E(g)$.

The antipodality-based estimator obtains 0.273 MAE and 0.704 AUC on the held-out test set of grasps. Although this estimator outperforms the constant baseline, we hope to improve the estimate of $E(g)$ using additional features.

D. Regression using BIDMach

1) Random Forests: We experiment with BIDMach's implementation of the Random Forests algorithm for regression. The two main parameters to select are the number of trees and their depth. During preliminary trials, as tree depth increased, validation-set MAE decreased from its initial value of roughly 0.30 to a typical value of 0.22 to 0.23

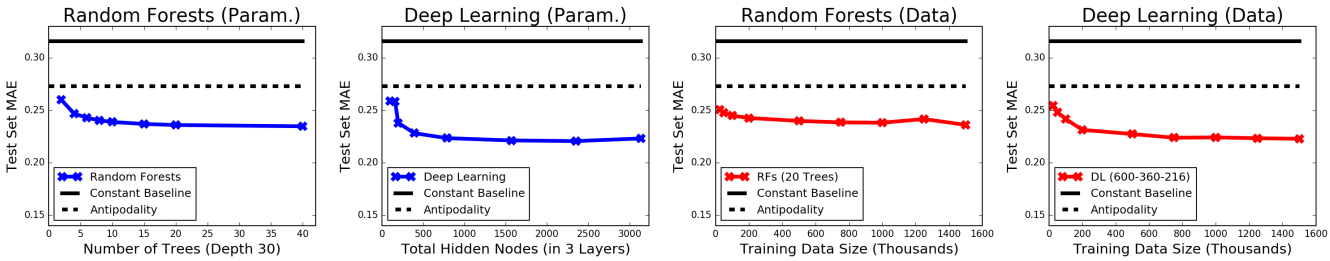

Fig. 6: MAE performance of Deep Learning and Random Forests on predicting grasp robustness as a function of the number of parameters (first two subplots) or the training data size (last two subplots). In the first two subplots, the algorithms were all trained on the full training data. We varied the Random Forests by increasing the number of trees, and Deep Learning by increasing the number of weights. (For DL, the subplot shows the total number of hidden nodes, so a 100-60-36 architecture is reported as having $100 + 60 + 36 = 196$ nodes.) In the last two subplots, the number of parameters in each model is fixed: 20 trees for RFs (each depth 30), and a 600-360-216 architecture for DL. We augment each subplot with the constant baseline from Section IV-B and the antipodality-based estimator from Section IV-C. Note that the y-axis scales are equivalent.

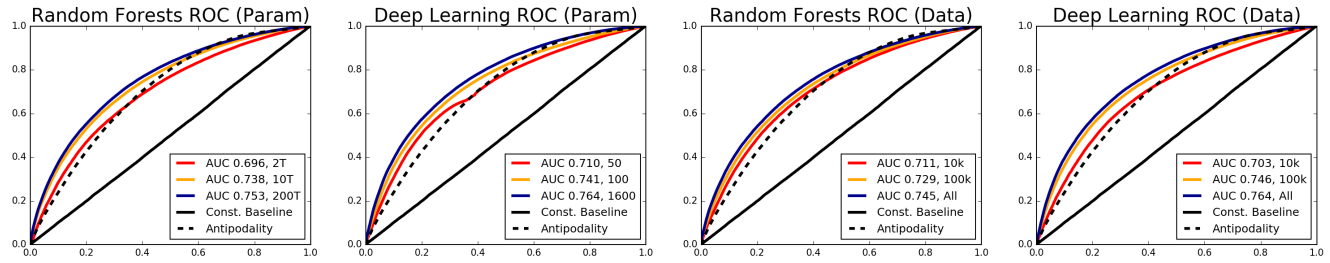

Fig. 7: AUC performance of Deep Learning and Random Forests on predicting grasp robustness as a function of the number of parameters (first two subplots) or the training data size (last two subplots). All four subplots show five ROC curves. In the first subplot, the number of trees are 2, 10, and 200, and in the second subplot, the number of nodes in the first hidden layer are 50, 100, and 1600 (using the standard 3 hidden layer, tapering weight design). In the last two subplots, we use 20 trees in the Random Forest and a 600-360-216 Deep Learning architecture. We augment each subplot with the ROCs for the constant baseline from Section IV-B and the antipodality-based estimator from Section IV-C. This figure must be viewed in color.

from depth 30 onward. At depth 30, BIDMach could train 20 trees at a time within our RAM constraints. More trees can be computed by running BIDMach's forest algorithm multiple times, potentially on different nodes since the trees are independent.

BIDMach's regression mode requires target values to be quantized, so we set the target granularity to 0.01 to match the input granularity. (Each $E(g)$ in our data is the number of successes divided by 100.) Thus, for a given grasp g as input, each tree generated from the algorithm will predict $E(g) \in \{0.0, 0.01, 0.02, \ldots, 0.99, 1.00\}$. Then the final estimated $E(g)$ is the average of the $E(g)$ estimates from all the trees.

2) Deep Learning: We use multilayer fully connected neural networks to learn the function $\hat{E}$ over the space of local patches.

We train the networks with BIDMach [15], on a single machine with two NVIDIA Titan X GPUs. We use stochastic gradient descent with ADAGrad [35] diagonal scaling. Deep networks of arbitrary topology can be constructed in BIDMach, but it also includes convenience functions that generate tapered (exponentially-decreasing layer size) multi-layer networks. We found these networks gave good performance compared to networks with arbitrary size in each layer, and provided a single parameter (the taper) to optimize over.

For all experiments, we use an optimized taper of $t = 0.6$. We apply L_1-regularization of the weights. The last layer applies the logistic loss function; it also has one unit, representing the predicted $E(g)$, while the input layer has 289 units. While the target value is not discrete, but a real value in $[0, 1]$, we can apply logistic loss by invoking the same rationale we used for ROC/AUC analysis.

We explored sigmoid, hyperbolic tangent, and rectified linear unit (ReLU) non-linearities between fully-connected layers in the network. We ran trials with a 600-360-216 neural network. For 500000 and 1 million training elements, both our AUCs and MAEs on the validation set were slightly better with the sigmoids compared to the hyperbolic tangents or ReLUs. Consequently, we use only sigmoids as activation functions. We also repeated these six experiments using dropout [36], but the AUC/MAE results did not improve so we do not use it.

To determine other details of the architecture, we ran preliminary trials with 500000 training cases while ranging the number of hidden layers from two to seven. For each layer count, we used 600 nodes in the first hidden layer and followed the tapering strategy for the remaining layers. Our best performance on the validation set was with three hidden layers (AUC 0.756, MAE 0.2256), though results were similar until seven hidden layers, where the network may have started to overfit. From these observations, we default to using three hidden layers (with sigmoids), resulting in a 600-360-216 architecture.

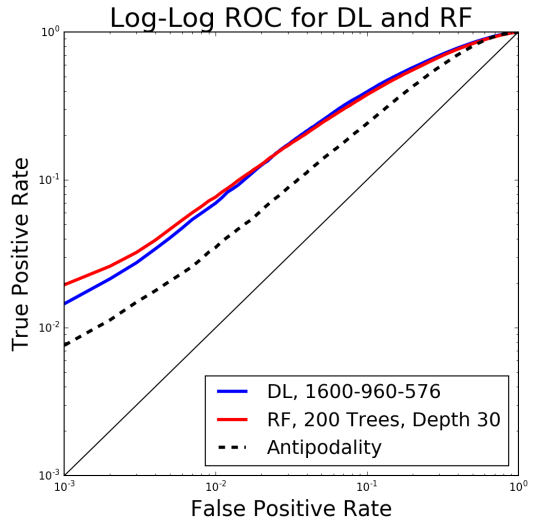

Fig. 8: A log-log plot representing the ROC curves for Random Forests using 200 trees, Deep Learning with a 1600-960-570 architecture, and the antipodality-based estimator, all trained on the entire training data. As outlined in Section IV-E.2, the log-log scale allows us to narrow our focus into the critical region of the graph to assess the likelihood of Type I errors. We see that Random Forests have slightly better guarantees in this region, though the difference may not be statistically significant.

3) Other Algorithms: We also experiment with three other algorithms in BIDMach's suite: (regularized) linear regression, logistic regression, and SVM (kernel) regression. We trained these each on the full training data using the standard minibatch framework. We increased the number of passes over the data up to the point when test-set MAE results stagnated to roughly 0.285 for all three algorithms. While this exceeds the constant baseline, the results are slightly worse with those of the antipodality predictor from Section IV-C and considerably worse than those from Deep Learning and Random Forests. Thus, we do not report further on linear, logistic, and SVM regression.

E. Performance Results

1) Mean Absolute Error (MAE): Fig. 6 shows the MAE of Random Forests and Deep Learning on the test set of 79000 grasps. The first two subplots show how performance varies based on the number of parameters (tree count for Random Forests, hidden nodes for Deep Learning) when training on the full training set. The last two subplots show the performance as a function of the training data size for a constant number of parameters (20 trees for Random Forests and a 600-360-216 architecture for Deep Learning). Fig. 6 is augmented to include the constant baseline from Section IV-B and the antipodality-based estimator from Section IV-C.

We see that Random Forests and Deep Learning substantially outperform the constant predictor and antipodality baselines across all parameter counts or training data sizes reported in Fig. 6. Comparing Deep Learning and Random Forests, we see that the former is slightly superior to the latter, with an edge of roughly 0.01 to 0.02 MAE once performance has leveled off.

Algorithm	Size (MB)	Train (sec)	Test (ms)
RF 20 trees, depth 30	427.53	1385	0.15
RF 200 trees, depth 30	4275.30	13850	0.21
DL 600-360-216, 10 passes	1.79	450	1.00
DL 1600-960-576, 10 passes	9.75	626	1.05

TABLE I: Four algorithms and associated model sizes (in MB), training time (seconds), and testing time (milliseconds) benchmarks. The training time is done with all the 1.55m training grasps, while the testing time is reported as the average of the prediction times for *one* grasp from the test set (of 79000 elements). For the Random Forest with 200 Trees, we trained ten separate 20 Tree Random Forests, so the training time is estimated to be a factor of ten larger.

2) Area Under the Curve (AUC) and Type I/II Errors: Fig. 7 is similar to Fig. 6, except that it uses ROC curves and AUC metrics instead of MAE. For readability, we only include three variations of Random Forests or Deep Learning in the subplots: each shows an ROC curve corresponding to a small amount of parameters (or data), a medium amount of parameters (or data), and a large amount of parameters (or data), which is useful to indicate when the algorithms exhibit diminishing returns. Overall, Fig. 7 provides similar conclusions as Fig. 6. The two best Deep Learning cases have an AUC of 0.764, which is slightly better than even our largest Random Forest with 200 trees (0.753 AUC) shown in the first subplot.

ROC curves allow us to analyze Type I and Type II errors. Here, a Type I error (false positive) is when one overestimates true robustness, and a Type II error (false negative) is when one underestimates true robustness. For a robot grasping an object, a Type I error is worse than a Type II error, because the former may result in dropping objects, but the latter may just cause the robot to overlook one of potentially many good grasps. In Fig. 7, we find that while the antipodality-based estimator is consistently superior to Deep Learning and Random Forests in certain regions of the ROC curve, it is consistently worse in the left portion, which is the more important area since that represents the desired low false positive rates.

Fig. 8 explores this in more detail using the 200 tree Random Forest and the highest performing Deep Learning model (with 0.764 AUC). The ROC curves are on a log-log scale to "magnify" the important space of the graph. Fig. 8 shows that Random Forests, despite having slightly lower overall AUC than Deep Learning, have slightly better performance in the low false positive rate region.

3) Timing and Memory: We now analyze the memory requirements and speed of our algorithms. Table I shows benchmarks for two representative Random Forests and Deep Learning models that achieve similar MAE and AUC results. We report on model size, training time, and testing time. The model size is based on the number of nodes in Random Forests (each 16 Bytes in BIDMach) or the number of weights in Deep Learning (each 4 Bytes in BIDMach). The training time is with the full training data; the Deep Learning models are trained over 10 full passes over the data, which is enough to cause validation set log-likelihood to converge.

As robots typically grasp one object in action, the reported

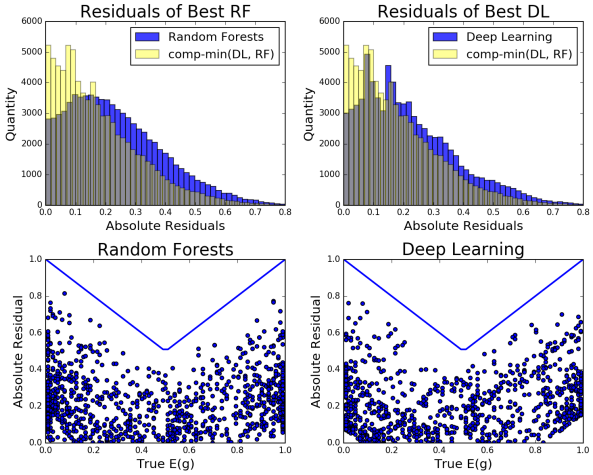

Fig. 9: Histograms and scatter plots for absolute residuals. (Top) Subplots representing counts of "absolute residual vectors" from the $E(g)$ predictions of the best DL and RF cases from Fig. 6 and Fig. 7. The component-wise minimum vector is overlaid on both histograms. The x-axes are cut off at 0.8 for readability. (Bottom) Scatter plots showing the absolute residuals of each learning algorithm as a function of the $E(g)$. A random subset of 1000 test data points is shown. The blue lines are an upper bound for the maximum possible absolute residual for a given $E(g)$ level.

testing (i.e., $E(g)$ prediction) time is the average time needed to evaluate $E(g)$ for one grasp across the 79000 testing grasps. To make comparisons fair, the predictions are applied on individual grasps at a time, and not done in batch mode.

Table I shows that Random Forests can take up far more memory than Deep Learning models to achieve comparable MAE/AUC results, but have faster evaluation speed with 0.15 and 0.21 milliseconds for RFs compared to 1.00 and 1.05 milliseconds for DL. These results makes sense because RFs predict using fast tree traversal, but the RF trees collectively require more memory than DL weight matrices. Dex-Net takes approximately 1.6 seconds to estimate a single $E(g)$. This means that Random Forests (using the 0.21ms value) and Deep Learning (using the 1.05ms value) are more than 7500x and 1500x faster, respectively, than Dex-Net.

F. Residuals and Failure Cases

As further exploration, for the best-performing Deep Learning and Random Forest models from Fig. 6 and Fig. 7, we form a vector of the predicted $E(g)$ values on the 79000 testing grasps. Then, we take the absolute difference of these vectors with the corresponding vector of true $E(g)$ values, $\langle E(g_1), E(g_2), \ldots, E(g_{79000})\rangle$, forming a vector of the "absolute residuals." The top half of Fig. 9 plots histograms of these absolute residuals and for each, overlays a third histogram corresponding to the vector containing the component-wise minimums. The intuition is that the "comp-min" histogram represents a baseline for how well *both* algorithms can identify $E(g)$ for a particular g; high values indicate grasp feature vectors that are challenging.

The grasps corresponding to the points with large absolute residuals in the comp-min vector are, by this particular measure, among the most difficult grasps to predict $E(g)$.

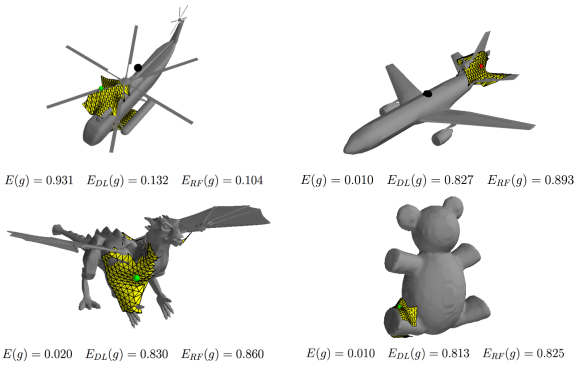

Fig. 10: Four patch and object meshes representing "failure cases," or those test-set grasps in which both Deep Learning and Random Forests poorly predicted $E(g)$. Starting at the top left and proceeding clockwise, the object meshes are a chopper, a plane, a bear, and a dragon. The true $E(g)$ values are listed below each grasp and mesh, along with the value predicted by Deep Learning ($E_{DL}(g)$) and Random Forests ($E_{RF}(g)$).

In Fig. 10, we plot four of these "failure cases." The dragon, bear, and plane grasps represent Type I errors by both algorithms, while the fourth (on a chopper) is a Type II error. From these samples, we observe certain characteristics that might intuitively indicate a failure case, such as patches that touch steady locations of an object but on uneven or thin surfaces (the chopper) or patches on smoother surfaces but which are far from the center of mass (the bear).

We further analyze the absolute residuals as a function of the target $E(g)$ values. In the bottom half of Fig. 9, we plot absolute residuals of 1000 random test data points. The scatter plots show that the majority of points have small residuals. They also have a rough "kink" in the center because the error is bounded by a piecewise linear function given by $f(x) = 1 - x$ if $x \in [0, 0.5)$ and $f(x) = x$ if $x \in [0.5, 1]$. The scatter plots also demonstrate that the residuals do not display unexpected behavior, a result that also holds during our re-runs with different subsets of points. Apart from the unavoidable kink in the center, the residuals thus do not contain extra information that our model misses.

V. Conclusions and Future Work

These large-scale supervised learning methods are substantially faster alternatives to Monte Carlo sampling for estimating grasp robustness. Overall, the results support the hypothesis that surface patch information can be used in both RF and DL methods to rapidly infer grasp robustness. In future work we will explore performance with other grasp quality metrics such as Ferrari-Canny [1] and those surveyed in [37], [38]. We will also investigate what happens when using other patch models and when object shape varies. Finally, we will explore how a fast robustness estimator can be used in the inner loop for a real robot's grasp planner that would take as input point clouds, so as to allow fast grasping in practice.

ACKNOWLEDGMENTS

This research was performed at UC Berkeley in affiliation with BID, AUTOLAB, AMP Lab, BAIR, and the CITRIS "People and Robots" (CPAR) Initiative for Information Technology in the Interest of Society: http://robotics.citris-uc.org. The authors were supported in part by the U.S. National Science Foundation under NRI Award IIS-1227536: Multilateral Manipulation by Human-Robot Collaborative Systems, and by Google, Cisco, Siemens, Cloudminds, UC Berkeley's Algorithms, Machines, and People Lab, the Berkeley Fellowship, the National Physical Science Consortium Fellowship, and the Knut & Alice Wallenberg Foundation. We thank Sanjay Krishnan for helpful feedback and suggestions.

REFERENCES

[1] C. Ferrari and J. Canny, "Planning optimal grasps," in *Proc. IEEE Int. Conf. Robotics and Automation (ICRA)*. IEEE, 1992.

[2] A. Bicchi and V. Kumar, "Robotic grasping and contact: A review," in *Proc. IEEE Int. Conf. Robotics and Automation (ICRA)*. IEEE, 2000.

[3] J. Mahler, S. Patil, B. Kehoe, J. van den Berg, M. Ciocarlie, P. Abbeel, and K. Goldberg, "Gp-gpis-opt: Grasp planning under shape uncertainty using gaussian process implicit surfaces and sequential convex programming," in *Proc. IEEE Int. Conf. Robotics and Automation (ICRA)*. IEEE, 2015.

[4] M. Laskey, J. Mahler, Z. McCarthy, F. T. Pokorny, S. Patil, J. van den Berg, D. Kragic, P. Abbeel, and K. Goldberg, "Multi-armed bandit models for 2d grasp planning with uncertainty," in *Proc. IEEE Conf. on Automation Science and Engineering (CASE)*. IEEE, 2015.

[5] G. Biau, "Analysis of a random forests model," *J. Mach. Learn. Res.*, vol. 13, no. 1, Apr. 2012.

[6] L. Breiman, "Random forests," *Mach. Learn.*, vol. 45, no. 1, pp. 5–32, Oct. 2001.

[7] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in Neural Information Processing Systems 25*, F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, Eds. Curran Associates, Inc., 2012, pp. 1097–1105.

[8] K. He, X. Zhang, S. Ren, and J. S. 0001, "Delving deep into rectifiers: Surpassing human-level performance on imagenet classification." *ICCV*, vol. abs/1502.01852, 2015.

[9] A. Y. Hannun, C. Case, J. Casper, B. C. Catanzaro, G. Diamos, E. Elsen, R. Prenger, S. Satheesh, S. Sengupta, A. Coates, and A. Y. Ng, "Deep speech: Scaling up end-to-end speech recognition," *arXiv preprint*, vol. abs/1412.5567, 2014.

[10] G. E. Hinton, L. Deng, D. Yu, G. E. Dahl, A. Mohamed, N. Jaitly, A. Senior, V. Vanhoucke, P. Nguyen, T. N. Sainath, and B. Kingsbury, "Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups," *IEEE Signal Process. Mag.*, vol. 29, no. 6, 2012.

[11] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski, S. Petersen, C. Beattie, A. Sadik, I. Antonoglou, H. King, D. Kumaran, D. Wierstra, S. Legg, and D. Hassabis, "Human-level control through deep reinforcement learning," *Nature*, vol. 518, no. 7540, pp. 529–533, 02 2015.

[12] D. Silver, A. Huang, C. J. Maddison, A. Guez, L. Sifre, G. van den Driessche, J. Schrittwieser, I. Antonoglou, V. Panneershelvam, M. Lanctot, S. Dieleman, D. Grewe, J. Nham, N. Kalchbrenner, I. Sutskever, T. Lillicrap, M. Leach, K. Kavukcuoglu, T. Graepel, and D. Hassabis, "Mastering the game of go with deep neural networks and tree search," *Nature*, vol. 529, no. 7587, pp. 484–489, 01 2016.

[13] M. Li, Y. Bekiroglu, D. Kragic, and A. Billard, "Learning of grasp adaptation through experience and tactile sensing," in *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems, Chicago, IL, USA, September 14-18, 2014*, 2014, pp. 3339–3346.

[14] J. Mahler, F. T. Pokorny, B. Hou, M. Roderick, M. Lasekey, M. Aubry, K. Kohlhoff, T. Kroeger, J. Kuffner, and K. Goldberg, "Dex-net 1.0: A cloud-based network of 3d objects and a multi-armed bandit model with correlated rewards to accelerate robust grasp planning," in *Proc. IEEE Int. Conf. Robotics and Automation (ICRA)*. IEEE, 2016.

[15] J. Canny and H. Zhao, "BIDMach: Large-scale learning with zero memory allocation," in *BigLearning, NIPS Workshop*, 2013.

[16] B. Siciliano and O. Khatib, *Springer handbook of robotics*. Springer Science & Business Media, 2008.

[17] K. Y. Goldberg and M. T. Mason, "Bayesian grasping," in *Proc. IEEE Int. Conf. Robotics and Automation (ICRA)*. IEEE, 1990.

[18] K. Goldberg, "Stochastic plans for robotic manipulation," Ph.D. dissertation, Carnegie Mellon University, 1990.

[19] V. N. Christopoulos and P. R. Schrater, "Grasping objects with environmentally induced position uncertainty," *PLoS Comput Biol*, vol. 5, no. 10, 2009.

[20] J. Kim, K. Iwamoto, J. J. Kuffner, Y. Ota, and N. S. Pollard, "Physically-based grasp quality evaluation under uncertainty," in *Proc. IEEE Int. Conf. Robotics and Automation (ICRA)*. IEEE, 2012.

[21] J. Weisz and P. K. Allen, "Pose error robust grasping from contact wrench space metrics," in *Proc. IEEE Int. Conf. Robotics and Automation (ICRA)*. IEEE, 2012.

[22] M. A. Roa and R. Suárez, "Finding locally optimum force-closure grasps," *Robotics and Computer-Integrated Manufacturing*, vol. 25, no. 3, 2009.

[23] A. T. Miller and P. K. Allen, "Graspit! a versatile simulator for robotic grasping," *Robotics & Automation Magazine, IEEE*, vol. 11, no. 4, 2004.

[24] F. T. Pokorny, K. Hang, and D. Kragic, "Grasp moduli spaces." in *Robotics: Science and Systems*, 2013.

[25] R. Detry, C. H. Ek, M. Madry, J. Piater, and D. Kragic, "Generalizing grasps across partly similar objects," in *Proc. IEEE Int. Conf. Robotics and Automation (ICRA)*. IEEE, 2012.

[26] D. Kappler, J. Bohg, and S. Schaal, "Leveraging big data for grasp planning," in *Proc. IEEE Int. Conf. Robotics and Automation (ICRA)*. IEEE, 2015.

[27] I. Lenz, H. Lee, and A. Saxena, "Deep learning for detecting robotic grasps," *IJRR*, 2015.

[28] C. Goldfeder and P. K. Allen, "Data-driven grasping." *Auton. Robots*, vol. 31, no. 1, 2011.

[29] I. Lenz, R. Knepper, and A. Saxena, "Deepmpc: Learning deep latent features for model predictive control," in *Proc. Robotics: Science and Systems (RSS)*, 2015.

[30] L. Pinto and A. Gupta, "Supersizing self-supervision: Learning to grasp from 50k tries and 700 robot hours," in *Proc. IEEE Int. Conf. Robotics and Automation (ICRA)*. IEEE, 2016.

[31] B. Kehoe, S. Patil, P. Abbeel, and K. Goldberg, "A survey of research on cloud robotics and automation," *IEEE Transactions on Automation Science and Engineering*, vol. 12, no. 2, 2015.

[32] J. Canny, H. Zhao, B. Jaros, Y. Chen, and J. Mao, "Machine learning at the limit," in *2015 IEEE International Conference on Big Data, Big Data 2015, Santa Clara, CA, USA*, 2015.

[33] R. M. Murray, S. S. Sastry, and L. Zexiang, *A Mathematical Introduction to Robotic Manipulation*, 1st ed. CRC Press, Inc., 1994.

[34] B. Li, Y. Lu, C. Li, A. Godil, T. Schreck, M. Aono, M. Burtscher, Q. Chen, N. K. Chowdhury, B. Fang, H. Fu, T. Furuya, H. Li, J. Liu, H. Johan, R. Kosaka, H. Koyanagi, R. Ohbuchi, A. Tatsuma, Y. Wan, C. Zhang, and C. Zou, "A comparison of 3d shape retrieval methods based on a large-scale benchmark supporting multimodal queries." *Computer Vision and Image Understanding*, vol. 131, 2015.

[35] J. Duchi, E. Hazan, and Y. Singer, "Adaptive Subgradient Methods for Online Learning and Stochastic Optimization," *J. Mach. Learn. Res.*, vol. 12, pp. 2121–2159, Jul. 2011.

[36] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: A simple way to prevent neural networks from overfitting," *Journal of Machine Learning Research*, vol. 15, 2014.

[37] R. Suárez, J. Cornella, and M. R. Garzón, *Grasp quality measures*. Institut d'Organització i Control de Sistemes Industrials, 2006.

[38] R. Balasubramanian, L. Xu, P. D. Brook, J. R. Smith, and Y. Matsuok, "Human-guided grasp measures improve grasp robustness on physical robot," in *Proc. IEEE Int. Conf. Robotics and Automation (ICRA)*. IEEE, 2010.