6.857 Final Project: Milestone 5

Sebastiani Aguirre Navarro and Rachel Holladay

I. Introduction

Our goal is to use neural networks to classify whether a particular grasp will succeed on an object. We utilize the Dexerity Network (DexNet) 2.0 data set [1], that has 6.7 million synthetic point clouds with parallel-jaw grasps (a common robot hand type of two parallel fingers) and analytical grasp metrics. The authors of the data set trained a Grasp Quality Convolutional Neural Network (GQ-CNN), which achieved 85.7% accuracy on their classification task. To accomplish the same task, we will be experimenting with new architectures, intput formats, other modifications described in Sec. IV. Most of the recent machine learning papers in robotics present a problem, dataset and, usually, an optimized convolutional neural network with some architecture and input format. Our goal is to explore the process of finding that CNN and exploring the factors that effect performance. While our results will only be verified according to this data set, and therefore cannot be generalized to all CNNs, we hope to gain intution, understanding, and, hopefully, a higher accuracy. Having explored various components, we will optimize our final, best architecture.

We will first describe the data set generation process and the features provided in the data set Sec. II. Understanding and processing this data set has become a larger element of our project then previously anticipated. We next discuss our results thus far Sec. III, which are preliminary. We will continue to explore these results, as well as our research questions Sec. IV.

II. Data Set

We are using the Dex Net 2.0 data set as first presented in [1]. We first briefly summarize their data generation process before describing how we manipulated the data.

Mahler et al define a generative graphical model defined over the camera pose, object shape and pose, friction coeffient, grasp, depth image and success metric. To generate the data set they make i.i.d (independent and identically distributed) samples from their generative graphical model, resulting in 6.7 million data points.

The data set is defined over 1,500 object meshes that were used in Dex-Net 1.0 [2], collected from a variety of other data bases and standardized with respect to position. For each object, they generated 100 parallel jaw grasps via rejection sampling of antipodal pairs and evaluated a grasp metric on each grasp. Additionally, each object is paired with a rendered depth image (2.5D point cloud [1]) from the sampled camera pose.

The GQ-CNN takes two images as input. The first is the depth image, called the "aligned image", transformed to center and axis align according the grasp point. Hence this image captures the scene and grasp in one representation. The second image, the "z image" is untransformed and represents the distance from the gripper to the camera.

The data set of 6.7 million data points has 21.1% positive examples. This is unsurprising, since it is much more difficult to find successful grasps, as compared to failed grasps.

The published Dex-Net 2.0 data set contains both sets of images for each data point in addition to grasp quality metrics and the grasp, represented by a 7-dimensional vector, specifying details of the grasp center, angle, object center and gripper width and several over parameters. Our label is given by the robust epsilon quality grasp metric (defined in [3]), which is thresholded by the value 0.002 to create binary labels.

From the 6.7 million data points, we create two types of data sets:

- **Unbalanced.** We randomly sample 10,000 data points from our entire set. We expect to sample approximately 20% positive examples, matching the distribution of the original set.
- **Balanced.** We randomly sample data points until we have 10,000 data points that are 50% positive examples and 50% negative examples.

We further discuss the motivation for this distinction in Sec. IV. For all data sets we include all possible features, although some architectures might not leverage all features.

Since we are sampling our data sets, we will sample multiple copies and average the final results across each version [2].

III. Results

During these stages of development, we are only evaluating our results according to the validation set. Since we are not yet achieving good enough performance on the validation set and are not doing parameter selection, we do not yet need to evaulate on the testing set.

[1] The images are 2D matrics that are referred to as 2.5D in robotics literature because they display depth information.

[2] This was not done for this milestone, but will be done in the final report.

For the following results, we use the balanced dataset with our input as the image for each data point and the 7-dimesional grasp vector. During training, 80% of the dataset was used as train set and the remaining 20% as test set. Below we describe and show the results of two achitectures, which we refer to as the Inception Network [4] and the ResNet. As discussed in Sec. IV, these are the some of the many networks we will be testing.

The inception network consists of 1 convolution layer in the beginning with 10 filters of size 3x3. The output of this layer is passed in parallel to three convolutional layers of sizes 1x1, 3x3, and 5x5, each with 16 filters. These outputs are concatenated on the depth dimension and passed through a max pooling layer of 3x3. The outputs are flattened with global average pooling and then the pose vector is concatenated before passed to a classifier of one hidden layer of 20 units, as seen in Fig. 3.

The residual network consists of two convolutional layers, one with 8 filters of size 7x7 and the next with 16 filters of size 3x3. At this point, the output of this layer branches, such that this same output is passed through two more convolution layers of 32 filters 3x3 and 16 filters 1x1 used as dimension reduction. The output of these two layers is added to their input and then passed to another convolution layer of 8 filters of 1x1 for further dimension reduction and then flattened with global average pooling. Like for the other network, the pose vector is concatenated to this output before passing it to a classifier with a fully connected layer of 10 hidden units, as shown in Fig. 4.

In Fig. 1, we can see that for the inception netowrk, the training loss decreases while the validation loss, while oscilating, increases. The same trend is shown in Fig. 2 for the residual net. This means that, like the results in our previous milestone, the network is overfitting to the data. The final training accuracy for inception net and residual net are 70% and 75% respectively, while both do 50% on the validation set. One possible modification is to add regularization on the fully connected layers, or to modify the architectures by removing or adding more layers. We will continue to explore this as well as experimenting with the representation of the data.

IV. Research Questions

Below we present several of the research questions we will continue to explore.

A. Input Format

One of the benefits of the Dex-Net data set is that it provides many features, allowing us to vary what we use as input to the network. We are and will continue to experiment with various inputs. Some inputs might be particularly informative but costly to collect when running the system on a real robot. Currently, the input

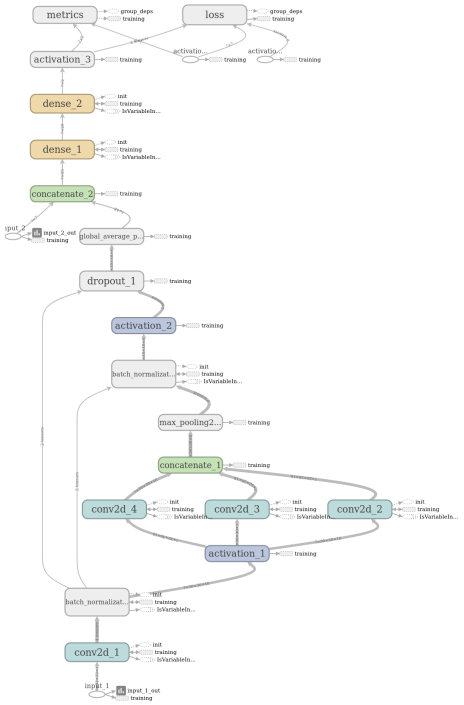


Fig. 3: Network Structure of Inception Net

format is a 32x32x1 depth map and a 1x7 pose vector. It is our intention to also use the representation used in the DexNet paper as soon as it becomes available [3]. This representation is as described in the Data Set section and shall serve as our baseline.

B. Balancing Data Sets

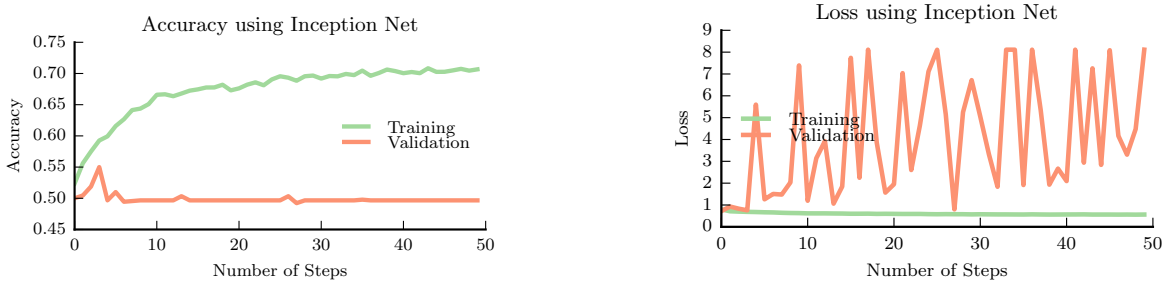
As mentioned previously, Dex-Net 2.0 contains approximately 20% positive examples. This is not inherenely problematic given that the training and testing sets are drawn from the same distribution, with this same ratio. However, by sampling subsets of our data set, we can achieve any positive-to-negative ratio and thus explore how changing this ratio effects accuracy. By doing this, we avoid a model to overfit and artificially think it is doing well by predicting one class on all samples. (For example, a very trival way to achieve 80% accuracy would be to predict negative for all examples.)

C. Data Set Size

The Dex-net data set contains 6.7 million data points. For computational reasons, we are sampling a subset of these points. However, we can vary the size of this subset to compare the trade-off between the accuracy and the size of the training set. Deep learning models tend to require a rather large quantity of data depending the classification task that it is being learned and how complex the data is.

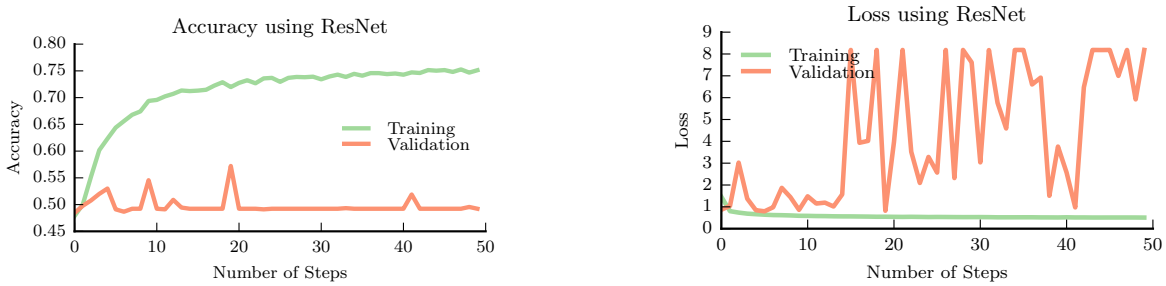
For now, since the input data consists of just small 32x32x1 images and 1x7 vectors, and we learning a

[3]We are coordinating with the authors of the paper



(a) Accuracy (b) Loss

Fig. 1: The Loss and Accuracy across the Training and Validation Sets for our Inception Net



(a) Accuracy (b) Loss

Fig. 2: The Loss and Accuracy across the Training and Validation Sets for our ResNet using the balanced data set.

binary classification task, we reasoned that we could start by using a small subset of the 6.7 million data points to produce a model of reasonable performance.

D. Architecture Structure

One of the largest sources of experimentation thus far and continuing forward is our choice of architecture. As our starting point, we produced an architecture that follows the concept of residual networks. This model consists of instead of learning a direct mapping of the input to the output, to learn "residue" over this input. Subsequently, we produced another architecture that follows the Inception model. Both were discribed in Sec. III and have batch normalization after each convolution, to mitigate internal covariances shifts, as well as a dropout rate of 0.7. This are initial architectures and not final. As we keep training and modifying them accordingly, we will come up with the proper parameters using the validation and testing accuracy as our metric.

E. Normalization

As seen in our results, we are falling prone to overfitting. Therefore, it is critical to explore how we can regularize the system.

References

[1] J. Mahler, J. Liang, S. Niyaz, M. Laskey, R. Doan, X. Liu, J. A. Ojea, and K. Goldberg, "Dex-net 2.0: Deep learning to plan robust grasps with synthetic point clouds and analytic grasp metrics," *arXiv preprint arXiv:1703.09312*, 2017.

[2] J. Mahler, F. T. Pokorny, B. Hou, M. Roderick, M. Laskey, M. Aubry, K. Kohlhoff, T. Kröger, J. Kuffner, and K. Goldberg, "Dex-net 1.0: A cloud-based network of 3d objects for robust grasp planning using a multi-armed bandit model with correlated rewards," in *ICRA*, pp. 1957–1964, IEEE, 2016.

[3] D. Seita, F. T. Pokorny, J. Mahler, D. Kragic, M. Franklin, J. Canny, and K. Goldberg, "Large-scale supervised learning of the grasp robustness of surface patch pairs," in *SIMPAR*, pp. 216–223, IEEE, 2016.

[4] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," in *CVPR*, pp. 1–9, IEEE, 2015.

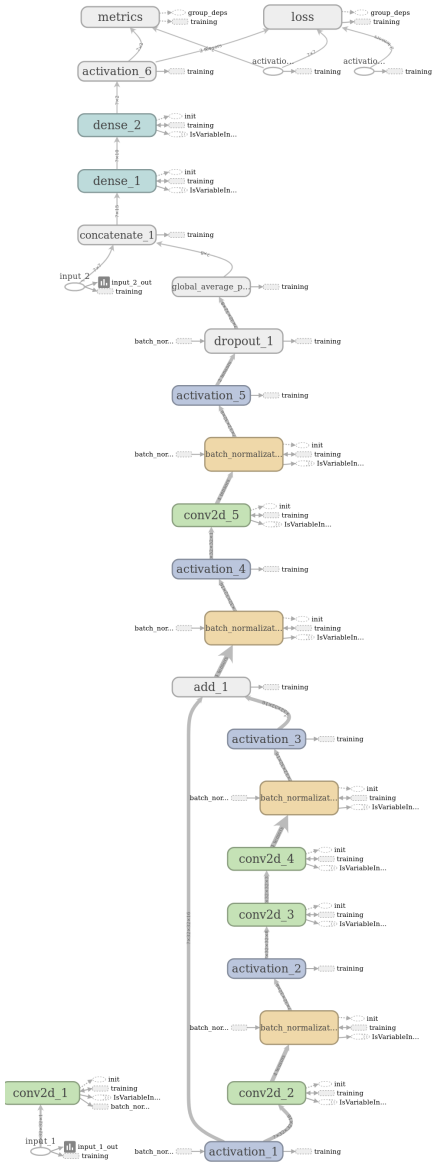


Fig. 4: Network Structure of ResNet