Visual Transfer Learning for Robot Manipulation

Contributor: Tianxiao He, Dongbing Han, Sheng Gao, Shengfeng Gu

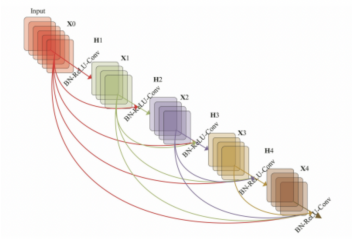
1. Introduction

1.1 Problem Definition

According to recent research, visual priors can assist in learning vision-based robot manipulation. In transfer learning, the model first learns a passive visual task and subsequently is active in a manipulation task. Here we revisit and analyze the importance of pre-learned visual representations in grasp detection tasks. We employed a pretrained DenseNet visual model and the Cornell grasping dataset and discovered that pre-training improves generalization and sample efficiency for object grasping prediction, especially for insufficient datasets.
Experiment codes can be found at https://github.com/hando189890/COMS6998-RoboticLearning

1.2 Model

Compared to AlexNet and ResNet, the Convolutional neural network DenseNet could connect all layers directly in the Dense Blocks framework. In DenseNet, each layer receives inputs from all preceding layers and transmits its feature maps to all subsequent layers [Figure 2]. It addresses the vanishing gradient issue and has been proven robust in multiple computer vision tasks. That is why we choose DenseNet as our model for grasp detection.



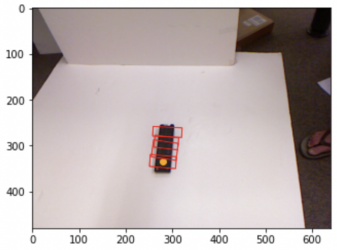


Fig. 1 The architecture of DenseNet. Fig. 2 & 3 Cornell Grasping Dataset with labeled grasps rectangle boxes.

1.3 Dataset

We use the Cornell Grasping Dataset, which has 885 pictures of 240 items with ground truth labeled grasps [Figure 2, 3]. Each image has numerous ground-truth positive and negative grasping rectangles with comprehensive labeling that vary in orientation, location, and scale. Due to such extensive labels for each image, this grasping dataset is optimal for our predicting task even though it is not comparative to recent synthetic datasets.

2. Approach

2.1 PVR framework and Training Process.

2.1.1 Pre-train visual models, Preprocess data and Data augmentation

In our project, we choose DenseNet121 from PyTorch as the pre-trained visual model [Figure 4].

According to our previous knowledge, pretraining large convolutional neural networks substantially reduces training time and is beneficial to manipulation tasks. Besides training DenseNet with RGB images, We also tested DenseNet with an RGB-D image dataset during the project, specifically by training from scratch. We modified the architecture to take the original image with depth channels as input. We wish to determine if combining RGB and depth information could improve grasping detection accuracy.

To preprocess image data, we resize PIL image to 256x256, preventing it from blurry and pixelated. We then crop the image to the center 224x224 pixels which gives an equal padding on both sides vertically and horizontally. After that, we convert the PIL image to a tensor in PyTorch (which also moves the channel dimension to the beginning). Last, we normalize the image by subtracting the mean and standard deviation of ImageNet.

To preprocess the label, we extract a 5 dimensional representation of grasp from the coordinates of four corners of the rectangles in the txt file. This new representation g = {x, y, theta, h, w}where (x,y) is the center of the rectangle, theta is the orientation of the rectangle to the horizontal axis of the image, h and w are the dimensions (height and width) of the rectangle. We write utility functions to convert between two formats for later uses.

Since our smaller image dataset is not as extensive as the recent and synthetic dataset, we make up for it by augmenting this dataset and expanding our image collection. The data augmentation method we use is flip left right and rotate by theta degree clockwise and counter-clockwise.

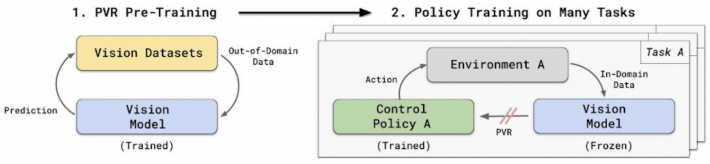


Figure 4, PVR architecture

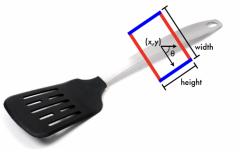


Figure 5, sample of rectangle evaluation metric for grasping task

2.1.2 Employ visual models to perform object-centric grasping task

Each of the models that we evaluated undergoes a similar training regimen. For all training processes, each model is trained for 50 epochs. After training, we undertake object-centric grasping tasks using visual models. Input for the object grasping task is either an RGB image or an RGB-D image, and the output is a grasping box. We also separated the dataset into 25%, 50%, 75%, and 100% of data. We incorporated different dataset sizes in all training processes to evaluate the effects of dataset size in the grasping detection performance for each model.

For ablation study, we want to check if transfer learning actually learned useful representation from out of domain data, and performs better then models that are trained from scratch. Therefore, we propose 3 models for training and evaluating in the robotic task:

- Frozen parameters : We take the pretrained model Densenet121 from pytorch library and load the weights pretrained on Imagenet. We freeze parameters from all previous layers in Densenet by setting their gradient update to False, then we append a new layer to map to our 5 dimensional grasping box.
- Fine tune pretrained model: Similarly, we load the weights pre trained on Imagenet for Densenet121. During training, we fine tune this model by allowing loss to backpropagate to

Densenet layers and update parameters from previous layers.

- Train from scratch: We randomly initialize weights for Densenet121 and train from scratch using robot grasping data

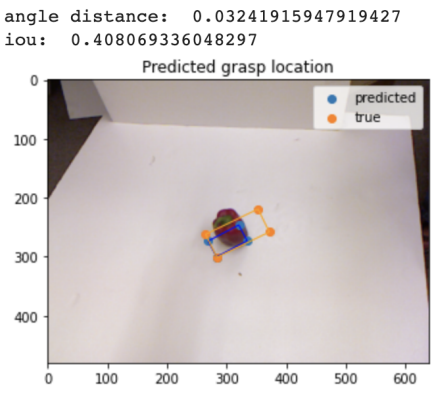
2.2 Evaluation Metrics

Among all 885 images of 240 distinct objects in the Cornell Grasping Dataset, We train DenseNet on 80% of the dataset and keep 20% for evaluation. We let the training set and the validation set not share the same image to reduce randomness. We evaluate the success rate of grasping both by the point metric and rectangle metric.

2.2.1 Point Metric

The point metric evaluates the distance between the center of the expected grasp and the center of each of the actual grasps. If any of these distances falls below a certain threshold (30 pixels), the grasp is deemed successful. This evaluation metric has several flaws, most notably not accounting for grasp angle or size. To be more serious, our project only includes the point metric as a result of comparing it to the rectangle metric.

2.2.2 Rectangle Metric



As shown in Figure 5 & 8, the rectangle metric considers a grasp to be success if both: 1) The grasp angle is within 30◦ of the ground truth grasp. 2) The interest over union of the predicted grasp and ground truth grasp is greater than iou_threshold (35%). The iou_threshold represents a value used in object detection to measure the overlap of a predicted versus actual bounding box for an object. The grasping, formally represented as g = {x, y, theta, h, w}which is mentioned above.

Fig. 8Example: predicted grasp location with grasping information

3. Result

3.1 Result without Cross Validation

We first examine the result without cross validation. The following Figure 6 and 7 shows grasping success rate with point/metric evaluation metric training from frozen parameters, fine tuning pretrained model or training from scratch with random initialization. The last column shows effects of depth information for improving prediction accuracy in gasping tasks. We evaluate results in different sizes of dataset used in training and prepare to see the patterns of how dataset size affects the training results.

The result indicates that, in general, as data size increases, the performance of all models evaluated using rectangle metrics improves. Frozen parameters perform better than fine-tuning for 50% and 75% of dataset size when evaluating with point metrics, but it's not obvious for 25% and 100% of

dataset. Random initialization performs poorly when the sample size of the data is small, but it outperforms other models as the sample size of the data increases.

When comparing the depth information in a model with the random initialization, it was evident that the point metric was more stable and could maintain a high success rate for all data sizes. Nevertheless, rectangle metrics are different. Overall, the performance of the model is enhanced by the depth information.

What's important is when comparing the frozen parameters with random initialization, where we could see a clear gap between the success rate of these two models during the 25% and 50% dataset. We conclude that transfer learning applied visual representations from data outside improves robot manipulation tasks, especially when data size is insufficient. However, the trend we observed is not entirely consistent with all models and data sizes. Thus, we incorporate cross validation for further investigation.

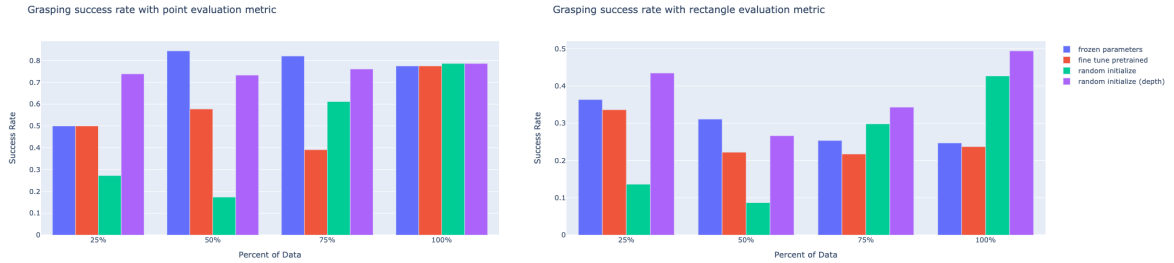


Figure 6 & 7 Grasping success rate with point/rectangle evaluation metric

3.2 Result with Cross Validation and Data augmentation

Cross-validation is used to protect a model from overfitting, especially if the amount of data available is limited. Here we introduce data augmentation and K-fold cross validation to further investigate the success rates with the parameter of dataset size in above three ablation studies.

Cross-validation results confirm the pattern demonstrated by non-cross-validation. The success rates evaluated by both point and rectangle metrics show an increase as the amount of the dataset increases. When the data size is insufficient, around 25%, frozen parameters, and fine-tuned pretrained cases show an outperformance in success rate compared to training from scratch with random initialization. The discovery confirmed that transfer learning applied visual representations from data outside improves robot manipulation tasks, especially when data size is insufficient without cross-validation.

In cross validation results, we see similar performance for frozen parameters and fine tune pretrained models. We examined the gradient and parameter update during training and confirmed that they are learning different representations. The reason that their performance is similar is because our evaluation metrics are not very sensitive to small changes in model output. Suppose two grasping boxes have slightly different positions and angles, yet they both succeed/fail to grasp certain objects given an input image. Given the small data size we have, it is possible that both evaluation metrics return the same success rate.

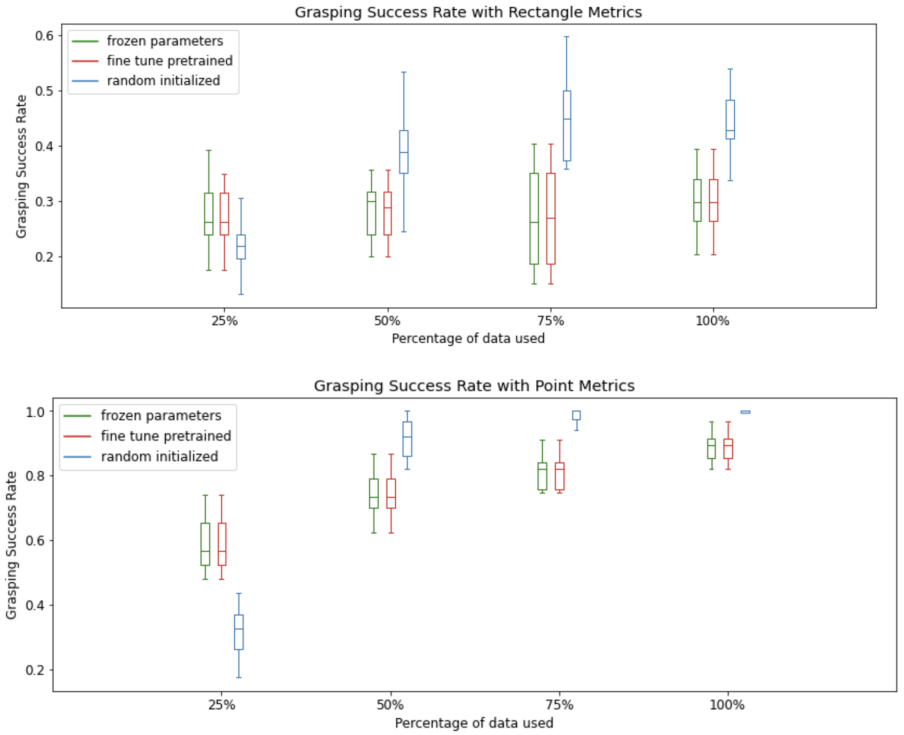


Figure 9 & 10 Cross-validated grasping success rate with point/rectangle evaluation metric

4. Takeaways and Conclusion

During the project, the biggest takeaway is that for training models on visual representation, it is essential to understand the details of the image dataset, such as data collection, data representation, data quality, and data visualization. Using the Cornell Grasping dataset, we originally trained the DenseNet model to identify single grasping points. However, the absence of a convergent training and validation loss curve indicates that the model is not learning. After that, when we attempted to visualize the grasping points of an object, we discovered that all grasping points are displayed parallel to the object's sides. Such discovery compelled us to investigate the dataset in greater depth. Consequently, we uncovered that the Cornell Grasping dataset could be more accurate when attempting to forecast grasping with bounding boxes.

The second takeaway is to evaluate the model performance, it is better to set up a fair baseline to find what works and what doesn't work. During our project, to evaluate the effects of pre-trained visual models on transfer learning with grasping manipulation tasks, we incorporated model training from scratch with random initialization. The results show that the success rate evaluated by rectangle metric of random initialization model not well-performed accounts to 25% and 50% dataset compared to pre-trained models with frozen parameters or fine tune pretrained model. Such results demonstrate that transfer learning is effective.

During the project, we demonstrate an efficient transfer of learning for predicting robotic object grasps from RGB/RGB-D images using DenseNet. Our models validated the observation that transfer learning applied to visual representations improves robot manipulation tasks, especially when data size is insufficient.