

EnsembleNet: Improving Grasp Detection using an Ensemble of Convolutional Neural Networks

Umar Asif
umarasif@au1.ibm.com

Jianbin Tang
jbtang@au1.ibm.com

Stefan Harrer
sharrer@au1.ibm.com

IBM Research
Melbourne, Australia

Abstract

Grasp detection from visual data is a recognition problem, where the goal is to determine regions in images which correspond to high grasp-ability with respect to certain quality metric. Existing deep learning based approaches mainly focus on predicting grasps, where the quality of the predictions is largely influenced by the choice of the CNN architecture and the objective function used for learning grasp representations. This paper presents a deep learning framework termed *EnsembleNet* which learns to produce and evaluate grasps within a unified framework. To achieve this, we formulate grasp detection as a two step procedure: i) Grasp generation - where, EnsembleNet generates four different grasp representations (regression grasp, joint regression-classification grasp, segmentation grasp, and heuristic grasp), and ii) Grasp evaluation - where EnsembleNet produces confidence scores for the generated grasps and selects the grasp with the highest score as the output. We evaluated EnsembleNet for grasp detection on RGB-D object datasets. The experiments show that the grasps produced by EnsembleNet are more accurate compared to the independent CNN models and the state-of-the-art grasp detection methods.

1 Introduction

Grasp detection is one of the fundamental problems in robotic grasping. A crucial challenge in grasp detection is generalization to novel objects in the presence of real-world challenges such as background clutter, variations in viewpoints, sensor noise, and scene complexity. With the advancements in deep learning, methods (e.g., [1, 2]) have shown significant improvements in grasp detection accuracy compared to analytical methods (e.g., [3]) which use hand-engineered feature learning to plan grasps from images or point clouds. In this context, one stream of work (e.g., [4, 5]) focuses on learning grasps in the form of a 2D rectangle (specified by its center position, width, height, and angle with the horizontal axis) from input images. Another stream of work (e.g., [6]) focuses on generating grasps by learning mapping between the images of the objects and robot's motion parameters, thereby allowing

the robot to iteratively refine its grasp target in the environment. These studies mainly focus on predicting grasps, where the accuracy of the predictions is largely influenced by the choice of the objective function or the CNN structure used to learn the grasps. Therefore, it is crucial to develop a mechanism to identify the CNN structure which learns the optimal mapping between the image data and the grasp representations. To achieve this, we present *EnsembleNet*, an ensemble of multiple CNN models which are trained to produce grasps using different objective functions (i.e., regression-only, joint regression-classification, and segmentation-based). The generated grasps are evaluated based on a quality metric to select the optimal grasp. Our hypothesis is that by ensembling multiple models (trained in different ways for a common task), our framework generates a higher performing model compared to the individual models. The main contributions of this paper are summarized below:

- 1) We propose “*EnsembleNet*” (Sec. 3), which combines multiple CNN structures and learns to produce and evaluate grasps within a unified framework. Experiments show that by combining models trained using different objective functions, our *EnsembleNet* produces more accurate grasps and achieves better generalization to novel objects compared to the individual models (see Sec. 4).
- 2) We propose “*SelectNet*”, a CNN model for grasp evaluation (Sec. 3.3). *SelectNet* learns grasp quality based on the orientation difference and intersection-over-union ratio between positive and negative grasp examples.

2 Related Work

Analytic grasp planning methods (e.g., [9]) follow a multi-stage pipeline where an input RGB-D image or a point cloud is processed through multiple stages (e.g., segmentation, object classification, and pose estimation). However, errors produced in the individual stages of the grasping pipeline significantly affect the overall grasp success. An alternate approach is to use deep learning to learn grasp representations in the form of 2D grasp rectangles [14] or 3D object poses [23] using RGB-D images or 3D point clouds. In this context, Lenz et. al. [14] created a database of RGB-D images annotated with human labels of good and bad grasps, and used the dataset to train CNN-based grasp detection models. The generation of a large annotated dataset requires significant time and human efforts. Therefore, alternative methods such as [12, 22] learned grasp representations using simulated images, and adapted the trained CNN models for real data. For instance, the work in [10] learned a CNN-based grasp detector using depth images rendered through dynamic simulations. Another stream of work learns visual-motor control by learning mapping between the images of the objects and robot’s motion parameters allowing the robot to iteratively refine its grasp target in the environment. For instance, the work in [17] presented another approach to large-scale data collection by recording more than 40k grasping trials on a real robot and used the data to train CNN models to predict grasping success. Recently, the work by Levine et al. [16] presented an approach to predict end effector poses by learning a prediction CNN using over 800k motor commands collected by multiple robotic arms continuously performing trial-and-error experiments for over 700 hours.

In this paper, we present a two-stage procedure to produce grasps using RGB-D images. **First**, our framework produces grasps using four different grasp representations (i.e., regression grasp, joint regression-classification grasp, segmentation grasp, and heuristics-based grasp). **Next**, given training examples of positive and negative grasps, our framework learns

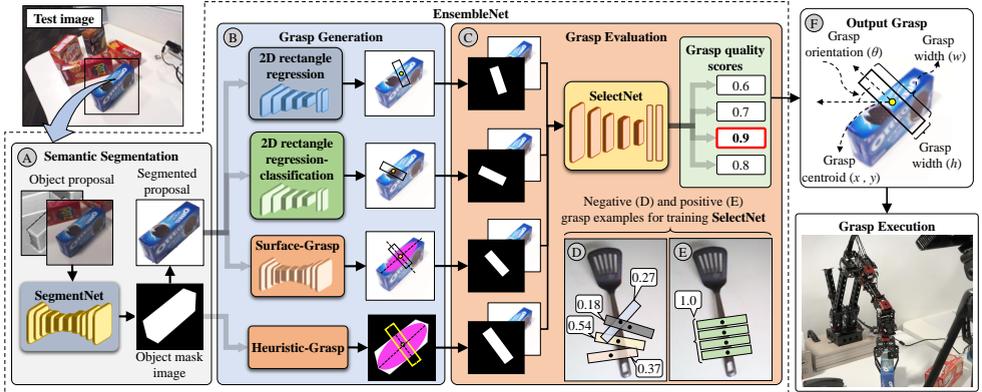


Figure 1: Given a 2D object proposal from a test image, our *EnsembleNet* uses a segmentation module (A) which refines the proposal by removing background information using a CNN for semantic segmentation. Next, it uses a cascade of four grasp generation models which compute grasps based on a regression loss, joint regression-classification loss, segmentation loss, and based on heuristics (B). Finally, *EnsembleNet* uses a grasp evaluation model termed “*SelectNet*” (C) which computes quality scores for the generated grasps based on a quality metric learned from negative and positive grasp examples shown in D and E, respectively. The grasp with the highest score is selected as the output for grasp execution.

to predict quality scores of the generated grasps. The grasp with the highest quality score is selected as the output.

3 The Proposed EnsembleNet

Fig. 1 shows the overall architecture of the proposed framework. It consists of three main modules: **i**) a segmentation module (Sec. 3.1), **ii**) a grasp generation module (Sec. 3.2), and **iii**) a grasp evaluation module (Sec. 3.3). In the following, we describe in detail, the different modules of the proposed framework.

3.1 Semantic Segmentation (Fig. 1-A)

The segmentation module takes an object proposal image as input and produces a segmentation mask which is used to refine the proposal (e.g., background removal). Let us denote the data structure of a single data sample as (I, \mathbf{Y}_g^o) , where $I \in \mathbb{R}^{N \times W \times H}$ represents the input image. The terms W , H and N represent the width, the height and the number of channels of the input image. The term $\mathbf{Y}_g^o \in \mathbb{R}^{W \times H}$ represents the ground-truth object segmentation mask. We use a CNN model based on an encoder-decoder architecture [49] which passes the input image through a number of convolution and deconvolution layers, and generates prediction maps of the size of the input image as shown in Fig. 2-A. The final layer is a deconvolution layer $DConvA$ which produces a pixel-wise labeling $\mathbf{Y}_{seg}^o \in \mathbb{R}^{n_c \times W \times H}$, where $n_c = 2$ represents the number of classes (object and background). The loss function for N_s labeled training images is given by:

$$\mathcal{L}_{seg} = \frac{1}{N_s} \sum_{i=1}^{N_s} \mathbf{S}_i, \quad (1)$$

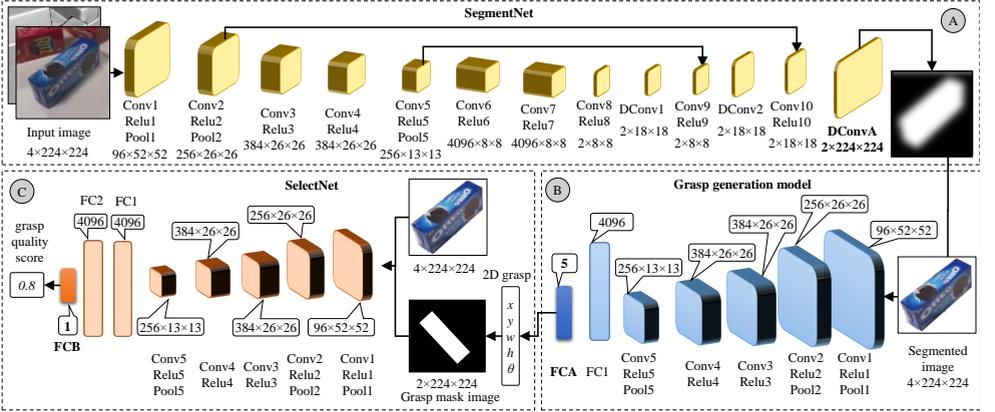


Figure 2: Detailed architecture of *EnsembleNet*. The segmentation model (A) is based on a convolution-deconvolution CNN structure and produces a segmented image. The grasp generation model (B) takes the segmented image as input and generates a grasp. The grasp parameters are used to generate a grasp mask image which is concatenated with the object image and fed to the grasp evaluation model (C) which generates a quality score. Note that for clarity we only show one grasp generation model in (B).

where $S_i \in \mathbb{R}^{(n_c \times W \times H)}$ represents the CNN normalized scores map for a sampled image i . It is computed by applying a SoftMax to the output of the *DConvA* layer of the network. Mathematically, S_i can be written as:

$$S_i = -\log \frac{e^{W_{y_i}^T f(x_i) + b_{y_i}}}{\sum_{j=1}^{n_c} e^{W_j^T f(x_i) + b_j}}, \quad (2)$$

where, $f(x_i)$ represents the output of the CNN layer for a data sample x_i . The terms y_i , W and b represent the class label, weights, and bias of the corresponding CNN layer, respectively.

3.2 Grasp Generation (Fig. 1-B)

The output of the segmentation module is fed to the grasp generation module which generates four different grasps (regression grasp, joint regression-classification grasp, segmentation grasp, and heuristics-based grasp) as shown in Fig. 1-B.

3.2.1 Regression Grasp

A regression grasp \mathbf{G}_{reg} is represented by an oriented rectangle in 2D image space defined by a 5-dimensional vector, given by:

$$\mathbf{G}_{reg} = [x, y, w, h, \theta], \quad (3)$$

where x and y represent the centroid of the rectangle as shown in Fig. 1-F. The term w and h represent the width and the height of the rectangle, respectively. The term θ represents the orientation of the rectangle given by the angle between the width of the rectangle and the x -axis. To learn regression grasps, we train a CNN model with an objective function given by:

$$\mathcal{L}_{reg} = \|\hat{\mathbf{G}}_{reg} - \mathbf{G}_{reg}\| / \|\mathbf{G}_{reg}\|_2, \quad (4)$$

where $\hat{\mathbf{G}}_{reg}$ represents the output of the last fully connected layer of the network (e.g., *FCA* as shown in Fig. 2), and \mathbf{G}_{reg} is a ground truth.

3.2.2 Joint Regression-Classification Grasp

Here we learn the parameters $[x, y, w, h]$ using a regression loss and quantize the orientation component θ into $N_\theta = 90$ classes (angular-bins), and learn θ using a CrossEntropy based objective function. Let $\mathbf{R}_i = [x, y, w, h]$ and θ_i denote the predicted values of the i^{th} image, respectively. We define the joint regression-classification loss function for the joint regression-classification grasp (\mathbf{G}_{joint}) over N_s images as:

$$\mathcal{L}_{joint}(\{\mathbf{R}_i, \theta_i\}_{i=1}^{N_s}) = \sum_i \left(\frac{\|\mathbf{R}_i - \mathbf{R}_i^*\|}{\|\mathbf{R}_i^*\|_2} \right) - \sum_i \theta_i \cdot \log(\theta_i^*), \quad (5)$$

where \mathbf{R}_i^* and θ_i^* represent the ground-truths.

3.2.3 Surface Grasp

A surface grasp is defined by a segmented region in the image as shown in Fig. 1-B. To generate surface grasps, we train a CNN model for pixel-wise labeling which produces prediction maps $\mathbf{Y}_{surf} \in \mathbb{R}^{n_c \times W \times H}$, where $n_c = 2$ represents the number of classes (grasp-region and background). The predictions map \mathbf{Y}_{surf} is max-pooled across the target classes to generate a binary grasp-region image. Next, we compute a grasp representation $\mathbf{G}_{surf} = [x, y, w, h, \theta]$, where x and y represent the centroid of the region. The width (w) and the height (h) parameters of the grasp are given by the length of the region along its minor axis and major axis, respectively, plus a threshold distance of 20 pixels. The orientation (θ) of the grasp is given by the angle between the x -axis and the major axis of the region. Note that the pixels labelled as grasp are considerably less compared to the background pixels. This results in large class imbalance in the training data which causes the learning process to trap in local minima of the objective function yielding predictions which are strongly biased towards background. To effectively deal with class imbalance, we used a weighted cross entropy loss as the objective function. Specifically, our loss function for N_s labeled training images is given by:

$$\mathcal{L}_g = -\frac{1}{N_s} \sum_{i=1}^{N_s} \sum_j \omega_j y_j \cdot \log(\hat{y}_j), \quad (6)$$

where, for a pixel location j , y is the ground truth label, \hat{y} is the network output, and ω is the weight of the correct class y applied in order to adjust the imbalance of pixel frequency across the target classes. The weight ω_k for a class k is computed as:

$$\omega_k = 1/\log(1.01 + f_k/f_T), \quad (7)$$

where, f_k represents the frequency of a class k , and f_T represents the sum of all class frequencies of the training set. The number 1.01 is an additional hyper-parameter which restricts the class weights to be in the interval of $[1, 100]$.

3.2.4 Heuristics-based Grasp

Heuristic grasp is generated from the output of the semantic segmentation network (Sec. 3.1). Specifically, the predictions map \mathbf{Y}_{seg}^o is max-pooled across the target classes to generate an object mask image as shown in Fig. 1-A. Next, we compute a grasp representation

$\mathbf{G}_{hrst} = [x, y, w, h, \theta]$, where x and y are given by the centroid location of the largest region in the mask image. The width and the height of the grasp are given by the length of the region along its minor axis and major axis, respectively, plus a threshold of 20 pixels. The orientation of the grasp is given by the angle between x-axis and the major axis of the region as shown in Fig. 1-B.

3.3 Grasp Evaluation

To select the best grasp out of the grasps produced by the grasp generation module, we introduce a CNN model termed “*SelectNet*” which learns to predict a quality score Q for an input grasp. Specifically, the input to *SelectNet* is the image of the object (\mathbf{I}), and a grasp mask image \mathcal{M} as shown in Fig. 2-C. To generate \mathcal{M} , we compute a 2D rectangle from the predicted grasp and use the four corners of the rectangle to generate a binary mask image as shown in Fig. 2-C. *SelectNet* is composed of five convolution layers and three fully connected layers. The image \mathbf{I} and its corresponding mask image \mathcal{M} are concatenated and fed to the model which produces a quality score as shown in Fig. 2-C. *SelectNet* is trained by minimizing an Euclidean loss on a training set which is comprised of correct and incorrect grasps. Specifically, each training image is labelled with a set of 10 correct grasps and a set of up to 20 incorrect grasps. The correct grasps are created from the ground truth grasp rectangles, where each grasp is assigned a quality score of 1.0. The incorrect grasps are generated by adding random shifts to the position and the orientation of the correct grasps with constraints to ensure coverage on the surface of the object. Let \mathcal{M}^- represents the mask image of an incorrect grasp rectangle for a query image, and $\mathbf{M} = \{\mathcal{M}_1^+, \dots, \mathcal{M}_i^+, \dots, \mathcal{M}_{N_g}^+\}$ represent the mask images of the correct grasps for the query image. The quality score of an incorrect grasp $Q(\mathcal{M}^-)$ is computed with respect to all the correct grasps (\mathbf{M}) of the query image, and the maximum of computed scores ($Q_1, \dots, Q_i, \dots, Q_{N_g}$) is assigned to \mathcal{M}^- . Mathematically, $Q(\mathcal{M}^-)$ can be written as:

$$Q(\mathcal{M}^-) = \max(Q_1(\mathcal{M}^-), \dots, Q_i(\mathcal{M}^-), \dots, Q_{N_g}(\mathcal{M}^-)), \quad (8)$$

$$Q_i(\mathcal{M}^-) = \frac{1}{\|\theta(\mathcal{M}_i^+) - \theta(\mathcal{M}^-)\|} \frac{|\mathcal{M}_i^+ \cap \mathcal{M}^-|}{|\mathcal{M}_i^+ \cup \mathcal{M}^-|}, \quad (9)$$

From Eq. 9 we observe that Q is high: **i)** if the orientation difference between an incorrect grasp and the correct grasp is low and, **ii)** if the overlap between the incorrect grasp and the correct grasp is high. Fig. 1- (D-E) show some example grasps with their corresponding quality scores. We train *SelectNet* with an objective function given by:

$$\mathcal{L}_{quality} = \|\hat{Q} - Q_g\| / \|Q_g\|_2, \quad (10)$$

where \hat{Q} represents the output of the last fully connected layer of the network (i.e., *FCB* in Fig. 2-C), and Q_g is a ground truth. During testing, each generated grasp is fed to *SelectNet* and a quality score is computed. The grasp with the highest quality score is selected as the output, given by:

$$G^* = \operatorname{argmax}_{k \in \mathcal{G}} Q_k, \quad (11)$$

where, $\mathcal{G} = [\mathbf{G}_{reg}, \mathbf{G}_{joint}, \mathbf{G}_{surf}, \mathbf{G}_{hrst}]$ is the set of grasps generated by EnsembleNet.

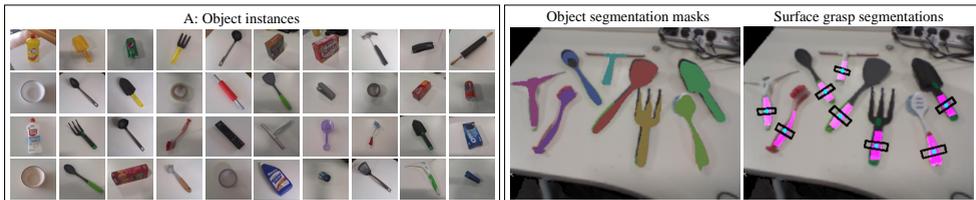


Figure 3: GraspSeg dataset contains RGB-D images of 42 objects (A). The ground truth is available in terms of object segmentations, grasp rectangles, and grasp segmentations.

3.4 Training and Implementation

We trained the segmentation, the grasp generation, and the grasp evaluation models independently. For this, we initialized the weights of the common layers of the networks using the weights pre-trained over ImageNet, and adjusted the weights by fine-tuning the models independently over the target datasets. Specifically, we initialized the weights of the layers $Conv1, \dots, Conv10$ of SegmentNet, the weights of the layers $Conv1, \dots, FC1$ of the grasp generation model, and $Conv1, \dots, FC1$ of SelectNet, with the weights pre-trained on ImageNet, and fine-tuned the models for 300k iterations using a learning rate of 0.0001. The layers $DCowA, FCA, FC2,$ and FCB were initialized from zero-mean Gaussian distributions with a standard deviation of 0.01, and trained from scratch using a learning rate of 0.01 (which was divided by 10 at 50% and 75% of the total number of iterations) and a parameter decay of 0.0005 (on the weights and biases). Our implementation is based on the Caffe library [10]. Training was performed by Stochastic Gradient Descent (SGD) with a batch size of 16.

4 Experiments

4.1 Datasets

We evaluated EnsembleNet on the grasp segmentation dataset (GraspSeg) of [9]. It provides 33,188 RGB-D images of 42 objects categorized into 15 different classes. The dataset also contains 6896 test images of indoor scenes containing multiple objects placed in different layouts. The ground truth in GraspSeg is available in the form of pixel-wise annotations for object segmentations and ground-truth for 2D grasp rectangles and surface-based grasp segmentations as shown in Fig. 3. The dataset was generated using an extended version of the scene labeling framework of [9] and [10]. We also evaluated EnsembleNet for grasp detection on the popular Cornell grasp dataset [15], which contains 885 RGB-D images of 240 objects.

4.2 Evaluation Criteria

For evaluation, we used the object-wise splitting criteria used in [15] which splits the objects randomly into train and validation subsets (i.e., the training set and the validation set do not share any images from the same object). This splitting strategy evaluates how well the model generalizes to unknown objects (objects which were not seen by the model during training). For computing grasp accuracy, we used the “rectangle-metric” proposed in [10]. It considers a grasp to be correct if: **i**) the difference between the predicted grasp angle and the ground-truth is less than 30° , and **ii**) the Jaccard index of the predicted grasp and the ground-truth is

Table 1: Mean grasp detection accuracy (%) of our EnsembleNet for different CNN design choices compared to the independent models on the GraspSeg dataset.

Model	Regression (Reg.)	Joint	Surface	EnsembleNet (Reg. + Joint + Surface)	
	grasp	grasp	grasp	No-SegmentNet	SegmentNet
[] SqueezeNet	58.7	60.3	61.9	65.4	67.8
[] MobileNet	64.4	65.8	66.6	71.1	73.7
[] VGG-16	74.1	75.4	77.5	81.5	83.2
[] ResNet-50	78.2	79.7	80.5	84.3	86.6

The average accuracy for the heuristics-based grasp is 55%.

Table 2: Mean grasp accuracy of our EnsembleNet for different grasp representations on the GraspSeg dataset.

Method	Reg. grasp	Joint grasp	Surface grasp
EnsembleNet (MobileNet+VGG16+ResNet50)	79.7%	81.3%	82.2%

Table 3: Object-wise average grasp detection accuracy on the Cornell grasp dataset [].

Method	Accuracy (%)	Method	Accuracy (%)
[] Fast search	58.3	[] Deep learning	75.6
[] SqueezeNet (Reg.)	77.7	[] SqueezeNet (Joint)	78.3
[] VGG16 (Joint)	78.9	[] MobileNet (Joint)	82.5
[] VGG16 (Reg.)	86.1	[] MultiGrasp	87.1
[] ResNet50 (Reg.)	87.3	[] MobileNet (Reg.)	88.5
[] Deep ResNets	88.9	[] Hybrid-Net	89.1
[] ResNet50 (Joint.)	89.7	[] GraspNet	90.2
<i>EnsembleNet</i> ¹	91.2	<i>EnsembleNet</i> ²	93.7
<i>EnsembleNet</i> ³	92.6	-	

*EnsembleNet*¹ = MobileNet (Reg.) + VGG16 (Reg.) + ResNet50 (Reg.)

*EnsembleNet*² = MobileNet (Joint) + VGG16 (Joint) + ResNet50 (Joint)

*EnsembleNet*³ = MobileNet (Joint.) + VGG16 (Reg.) + ResNet50 (Joint.)

higher than 25%. The Jaccard index for a predicted grasp rectangle \mathcal{R}^* and a ground-truth grasp rectangle \mathcal{R}^g is defined as:

$$J(\mathcal{R}^g, \mathcal{R}^*) = |\mathcal{R}^g \cap \mathcal{R}^*| / |\mathcal{R}^g \cup \mathcal{R}^*|. \quad (12)$$

4.3 Results

First, we evaluate EnsembleNet by combining multiple grasp representations and compare the results with the individual models (i.e., regression (Reg.), joint regression-classification (Joint), and surface grasp models). Specifically, we evaluate EnsembleNet with four different model choices (SqueezeNet [], MobileNet [], VGG-16 [], and ResNet-50 []) as its grasp generation models. The results of this experiment are provided in Table 1 which shows that our EnsembleNet achieves the highest grasp accuracy for all the tested model choices. The heuristics-based grasp model produces grasps which are always located at the centroid of the object. Although, centroid-based grasps are valid for objects with simple rectangular

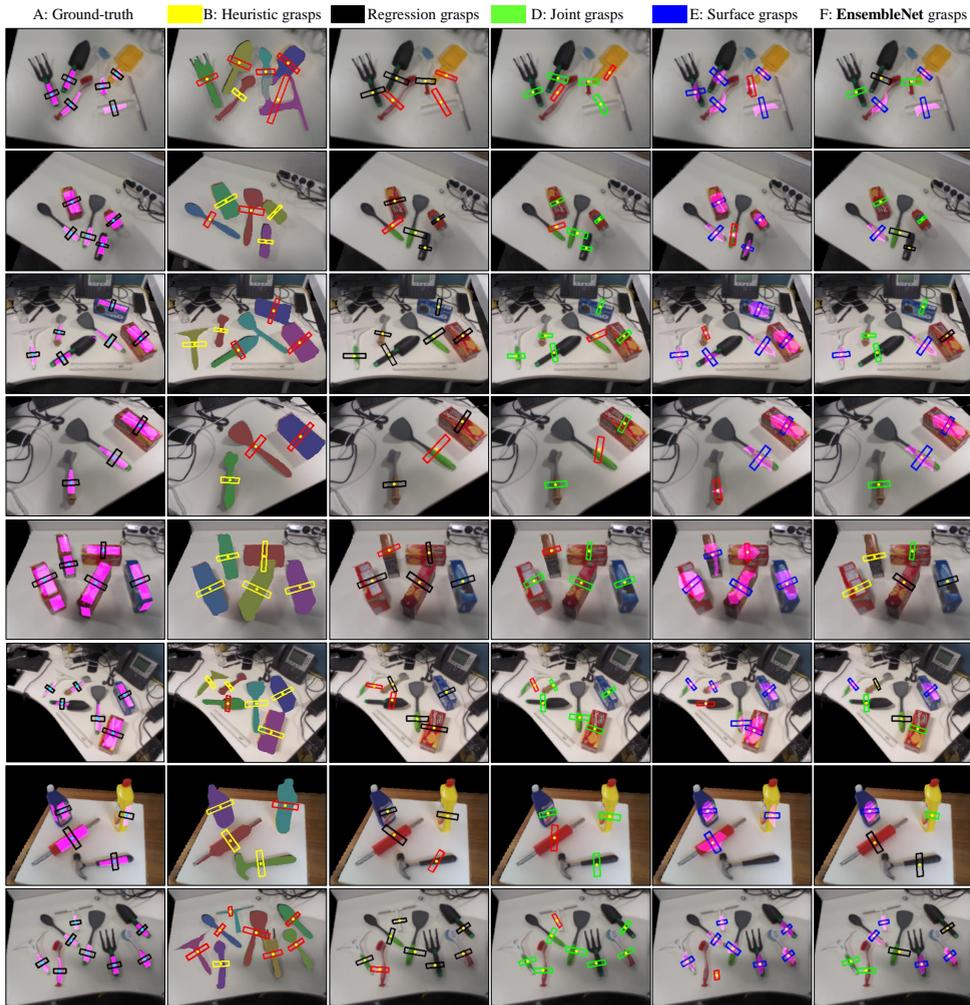


Figure 4: Qualitative results of grasp detection on the GraspSeg dataset produced by the proposed EnsembleNet and the individual grasp models (Heuristic grasps shown in yellow, regression grasps (Reg.) shown in black, joint regression-classification grasps (Joint) shown in green, and surface grasps shown in blue). Failure cases are highlighted in red rectangles.

or cylindrical shapes, they are invalid for objects with complex shapes (e.g., see the grasps highlighted in red in Fig. 4-B). From Table 1 we observe that on average, the surface grasps are superior than the regression or the joint grasps however, a careful investigation of the results reveals that there are certain situations where regression or joint grasps outperform the surface grasps. Fig. 4-E (highlighted in red) shows some failure cases of the surface grasp model. The corresponding results of the regression-only or the joint grasp models are correct as shown in Fig. 4-C and Fig. 4-D, respectively. For almost all of these cases, the surface grasp model failed to generate correct segmentations (shown in magenta in Fig. 4-E). The proposed EnsembleNet leverage the strengths of the different grasp representations by incorporating their corresponding trained CNN models into an ensemble architecture (Sec. 3), and learns to select the best grasp using the proposed grasp evaluation model (Sec. 3.3).

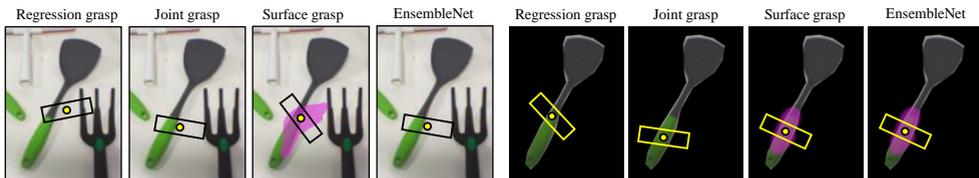


Figure 5: Qualitative comparison of the grasps produced without (left four images), and with (right four images) the proposed SegmentNet as a preprocessing step to grasp generation.

Table 1 also shows that the integration of the proposed semantic segmentation (Sec. 3.1) as a pre-processing step to grasp generation increases the grasp accuracy by almost 2% on average on the GraspSeg dataset. The proposed segmentation module refines the object proposals by removing background clutter as shown in Fig. 5.

Next, we evaluate EnsembleNet by combining different CNN structures (MobileNet, VGG16, and ResNet50) for grasp generation. Table 2 and Table 3 show the results of these experiments on the GraspSeg dataset and the Cornell grasp dataset, respectively. Table 2 shows that EnsembleNet produces improvements in the grasp detection accuracy for all the tested grasp representations compared to the individual CNN structures (shown in Table 1). This further validates that the proposed SelectNet (Sec. 3.3) enables EnsembleNet to select the optimal grasp (from the outputs of the individual models) based on the proposed quality metric (Sec. 9).

Table 3 shows a comparison of grasp accuracy produced by our EnsembleNet and other methods on the Cornell grasp dataset [14]. The results show that the grasp accuracy is strongly influenced by both the choice of the CNN structure and the objective function used for training the model. By combining different CNN structures with different objective functions for grasp generation, our model variants (e.g., *EnsembleNet*¹ or *EnsembleNet*²) learn different image-to-grasp mappings compared to the individual models, and yield superior accuracy compared to the state-of-the-art as shown in Table 3.

5 Conclusion and Future Work

We presented a deep learning based framework (termed EnsembleNet) to produce grasps by combining a heterogeneous collection of CNN models trained using different objective functions. Specifically, our EnsembleNet generates grasps from four different perspectives (i.e., regression, joint regression-classification, surface-based, and heuristics-based grasps) using RGB-D images. We also introduce a novel perspective of ensembling by evaluating the generated grasps in terms of grasp quality to select the best grasp out of the ensemble. In experiments we show that the proposed EnsembleNet is 2-5% more accurate compared to independent CNN models trained on regression, joint regression-classification, or surface grasp representations. In future, we plan to extend EnsembleNet for simultaneous object detection, classification, and grasp inference by integrating additional CNN branches into the cascaded architecture. We also plan to reduce the computational burden of the proposed EnsembleNet through parameter-pruning and using memory-efficient CNN architectures for real-time applications.

References

- [1] Umar Asif, Mohammed Bennamoun, and Ferdous Sohel. Simultaneous dense scene reconstruction and object labeling. In *ICRA*, pages 2255–2262. IEEE, 2016.
- [2] Umar Asif, Mohammed Bennamoun, and Ferdous Sohel. A multi-modal, discriminative and spatially invariant cnn for rgb-d object labeling. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2017.
- [3] Umar Asif, Mohammed Bennamoun, and Ferdous A Sohel. Rgb-d object recognition and grasp detection using hierarchical cascaded forests. *IEEE Transactions on Robotics*, 2017.
- [4] Umar Asif, Jianbin Tang, and Stefan Harrer. Graspnet: An efficient convolutional neural network for real-time grasp detection for low-powered devices. In *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI-18*, pages 4875–4882. International Joint Conferences on Artificial Intelligence Organization, 7 2018. doi: 10.24963/ijcai.2018/677. URL <https://doi.org/10.24963/ijcai.2018/677>.
- [5] Matei Ciocarlie, Kaijen Hsiao, Edward Gil Jones, Sachin Chitta, Radu Bogdan Rusu, and Ioan A Şucan. Towards reliable grasping and manipulation in household environments. In *Experimental Robotics*, pages 241–252. Springer, 2014.
- [6] Di Guo, Fuchun Sun, Huaping Liu, Tao Kong, Bin Fang, and Ning Xi. A hybrid deep architecture for robotic grasp detection. In *Robotics and Automation (ICRA), 2017 IEEE International Conference on*, pages 1609–1614. IEEE, 2017.
- [7] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, pages 770–778, 2016.
- [8] Andrew G Howard, Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto, and Hartwig Adam. Mobilenets: Efficient convolutional neural networks for mobile vision applications. *arXiv preprint arXiv:1704.04861*, 2017.
- [9] Forrest N Iandola, Song Han, Matthew W Moskewicz, Khalid Ashraf, William J Dally, and Kurt Keutzer. Squeezenet: Alexnet-level accuracy with 50x fewer parameters and 0.5mb model size. *arXiv preprint arXiv:1602.07360*, 2016.
- [10] Yangqing Jia, Evan Shelhamer, Jeff Donahue, Sergey Karayev, Jonathan Long, Ross Girshick, Sergio Guadarrama, and Trevor Darrell. Caffe: Convolutional architecture for fast feature embedding. *arXiv preprint arXiv:1408.5093*, 2014.
- [11] Yun Jiang, Stephen Moseson, and Ashutosh Saxena. Efficient grasping from rgbd images: Learning using a new rectangle representation. In *ICRA*, pages 3304–3311, 2011.
- [12] Edward Johns, Stefan Leutenegger, and Andrew J Davison. Deep learning a grasp function for grasping under gripper pose uncertainty. In *IROS*, pages 4461–4468. IEEE, 2016.
- [13] Sulabh Kumra and Christopher Kanan. Robotic grasp detection using deep convolutional neural networks. In *IROS*. IEEE, 2017.

-
- [14] Ian Lenz, Honglak Lee, and Ashutosh Saxena. Deep learning for detecting robotic grasps. *arXiv preprint arXiv:1301.3592*, 2013.
- [15] Ian Lenz, Honglak Lee, and Ashutosh Saxena. Deep learning for detecting robotic grasps. *IJRR*, 34(4-5):705–724, 2015.
- [16] Sergey Levine, Peter Pastor, Alex Krizhevsky, Julian Ibarz, and Deirdre Quillen. Learning hand-eye coordination for robotic grasping with deep learning and large-scale data collection. *IJRR*, page 0278364917710318, 2016.
- [17] Lerrel Pinto and Abhinav Gupta. Supersizing self-supervision: Learning to grasp from 50k tries and 700 robot hours. In *ICRA*, pages 3406–3413. IEEE, 2016.
- [18] Joseph Redmon and Anelia Angelova. Real-time grasp detection using convolutional neural networks. In *ICRA*, pages 1316–1322. IEEE, 2015.
- [19] Evan Shelhamer, Jonathan Long, and Trevor Darrell. Fully convolutional networks for semantic segmentation. *PAMI*, 39(4):640–651, 2017.
- [20] Karen Simonyan and Andrew Zisserman. Two-stream convolutional networks for action recognition in videos. In *Advances in neural information processing systems*, pages 568–576, 2014.
- [21] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.
- [22] Matthew Veres, Medhat Moussa, and Graham W Taylor. Modeling grasp motor imagery through deep conditional generative models. *IEEE Robotics and Automation Letters*, 2(2):757–764, 2017.
- [23] Andy Zeng, Kuan-Ting Yu, Shuran Song, Daniel Suo, Ed Walker, Alberto Rodriguez, and Jianxiong Xiao. Multi-view self-supervised deep learning for 6d pose estimation in the amazon picking challenge. In *ICRA*, pages 1386–1383. IEEE, 2017.