Densely Supervised Grasp Detector (DSGD)

Umar Asif
IBM Research Australia
umarasif@au1.ibm.com

Jianbin Tang
IBM Research Australia
jbtang@au1.ibm.com

Stefan Harrer
IBM Research Australia
sharrer@au1.ibm.com

Abstract

This paper presents *Densely Supervised Grasp Detector (DSGD)*, a deep learning framework which combines CNN structures with layer-wise feature fusion and produces grasps and their confidence scores at different levels of the image hierarchy (i.e., global-, region-, and pixel-levels). Specifically, at the global-level, DSGD uses the entire image information to predict a grasp. At the region-level, DSGD uses a region proposal network to identify salient regions in the image and uses a grasp prediction network to generate segmentations and their corresponding grasp poses of the salient regions. At the pixel-level, DSGD uses a fully convolutional network and predicts a grasp and its confidence at every pixel. During inference, DSGD selects the most confident grasp as the output. This selection from hierarchically generated grasp candidates overcomes limitations of the individual models. DSGD outperforms state-of-the-art methods on the Cornell grasp dataset in terms of grasp accuracy. Evaluation on a multi-object dataset and real-world robotic grasping experiments show that DSGD produces highly stable grasps on a set of unseen objects in new environments. It achieves 97% grasp detection accuracy and 90% robotic grasping success rate with real-time inference speed.

Introduction

Grasp detection is a crucial task in robotic grasping because errors in this stage affect grasp planning and execution. A major challenge in grasp detection is generalization to unseen objects in the real-world. Recent advancements in deep learning have produced Convolutional Neural Network (CNN) based grasp detection methods which achieve higher grasp detection accuracy compared to hand-crafted features. Methods such as (Lenz, Lee, and Saxena 2015; Redmon and Angelova 2015; Asif, Bennamoun, and Sohel 2017b; Asif, Tang, and Harrer 2018a) focused on learning grasps in a global-context (i.e., the model predicts one grasp considering the whole input image), through regression-based approaches (which directly regress the grasp parameters defined by the location, width, height, and orientation of a 2D rectangle in image space). Other methods such as (Pinto and Gupta 2016) focused on learning grasps at patch-level by extracting patches (of different sizes) from the image and predicting a grasp for each patch. Recently, methods such as (Morrison, Corke, and Leitner 2018; Zeng et al. 2017) used auto-encoders to learn grasp parameters at each pixel in the image. They showed that one-to-one mapping (of image data to ground truth grasps) at the pixel-level can effectively be learnt using small CNN structures to achieve fast inference speed. These studies show that grasp detection performance is strongly influenced by three main factors: **i)** The choice of the CNN structure used for feature learning, **ii)** the objective function used to learn grasp representations, and **iii)** the image hierarchical context at which grasps are learnt (e.g., global or local). In this work, we explore the advantages of combining multiple global and local grasp detectors and a mechanism to select the best grasp out of the ensemble. We also explore the benefits of learning grasp parameters using a combination of regression and classification objective functions. Finally, we explore different CNN structures as base networks to identify the best performing architecture in terms of grasp detection accuracy. The main contributions of this paper are summarized below:

1) We present *Densely Supervised Grasp Detector* (DSGD), an ensemble of multiple CNN structures which generate grasps and their confidence scores at different levels of image hierarchy (i.e., global-level, region-level, and pixel-level).

2) We propose a region-based grasp network, which learns to segment salient parts (e.g., handles or extrusions) from the input image, and uses the information about these parts to learn class-specific grasps (i.e., each grasp is associated with a probability with respect to a graspable class and a non-graspable class).

3) We perform an ablation study of our DSGD by varying its critical parameters and present a grasp detector that achieves real-time speed and high grasp accuracy.

4) We demonstrate the robustness of DSGD for producing stable grasps for unseen objects in real-world environments using a multi-object dataset and robotic grasping experiments. See our experiment videos at: https://youtu.be/Bn_dQ8vcNzs and https://youtu.be/tA2qgtbTT98

Related Work

In the context of deep learning based grasp detection, methods such as (Saxena, Driemeyer, and Ng 2008; Jiang, Moseson, and Saxena 2011; Lenz, Lee, and Saxena 2015) trained sliding window based grasp detectors. However, their high inference times limit their application for real-time systems. Other methods such as (Mahler et al. 2016; 2017; Johns, Leutenegger, and Davison 2016) reduced inference time by processing a discrete set of grasp candidates, but these methods ignore some potential grasps. Alternatively, methods such as (Redmon and Angelova 2015; Kumra and Kanan 2017; Guo et al. 2017) proposed end-to-end CNN-based approaches which regress a single grasp for an input image. However, these methods tend to produce average grasps which are invalid for certain symmetric objects (Redmon and Angelova 2015). Recently, multi-grasp detectors based on auto-encoders (Morrison, Corke, and Leitner 2018; Zeng et al. 2017; 2018; Myers et al. 2015; Varley et al. 2015) and Faster-RCNN (Ren et al. 2015) based grasp detector (Chu, Xu, and Vela 2018) demonstrated higher grasp accuracy compared to the global methods. Another stream of work focused on learning mapping between images of objects and robot motion parameters using reinforcement learning, where the robot iteratively refines grasp poses through real-world experiments (Pinto and Gupta 2016; Levine et al. 2016). In this paper, we present a grasp detector which has several key differences from the current grasp detection methods. **First**, our detector generates multiple global and local grasp candidates and selects the grasp with the highest quality. This allows our detector to effectively recover from the errors of the individual global or local models. **Second**, we introduce a region-based grasp network which uses segmentation information of salient parts of objects (e.g., handles, extrusions) to learn grasp poses, and produces more accurate grasp predictions compared to global (Kumra and Kanan 2017) or local detectors (Pinto and Gupta 2016). **Finally**, we use layer-wise dense feature fusion (Huang et al. 2017) within the CNN structures. This maximizes variation in the information flow across the networks and produces better image-to-grasp mappings compared to the models of (Redmon and Angelova 2015; Morrison, Corke, and Leitner 2018).

Problem Formulation

Given an image of an object as input, the goal is to generate grasps at different image hierarchical levels (i.e., global-, region- and pixel-levels), and select the most confident grasp as the output. We define the global grasp by a 2D oriented rectangle on the target object in the image space. It is given by:

$$\mathcal{G}_g = [x_g, y_g, w_g, h_g, \theta_g, \rho_g], \tag{1}$$

where x_g and y_g represent the centroid of the rectangle. The terms w_g and h_g represent the width and the height of the rectangle. The term θ_g represents the angle of the rectangle with respect to x-axis. The term ρ_g is *grasp confidence* and represents the quality of a grasp. Our region-level grasp is defined by a class-specific representation, where the parameters of the rectangle are associated with n classes (a graspable class: $n = 1$, and a non-graspable class: $n = 0$). It is given by:

$$\mathcal{G}_r = [x_r^n, y_r^n, w_r^n, h_r^n, \theta_r^n, \rho_r^n], n \in [0, 1]. \tag{2}$$

Our pixel-level grasp is defined as:

$$\mathcal{G}_p = [\boldsymbol{M}_{xy}, \boldsymbol{M}_w, \boldsymbol{M}_h, \boldsymbol{M}_\theta] \in \mathbb{R}^{s \times W \times H}, \tag{3}$$

where $\boldsymbol{M}_{xy}$, $\boldsymbol{M}_w$, $\boldsymbol{M}_h$, and $\boldsymbol{M}_\theta$ represent $\mathbb{R}^{s \times W \times H}-$dimensional heatmaps[1], which encode the position, width, height, and orientation of grasps at every pixel of the image, respectively. The terms W and H represent the width and the height of the input image respectively. We learn the grasp representations ($\mathcal{G}_g$, $\mathcal{G}_r$, and $\mathcal{G}_p$) using joint regression-classification based objective functions. Specifically, we learn the position, width, and the height parameters using a Mean Squared Loss, and learn the orientation parameter using a Cross Entropy Loss with respect to $N_\theta = 20$ classes (angular-bins).

The Proposed DSGD (Fig. 1)

Our DSGD is composed of four main modules as shown in Fig. 1: a base network for feature extraction, a Global Grasp Network (GGN) for producing a grasp at the image-level, a Region Grasp Network (RGN) for producing grasps using salient parts of the image, and a Pixel Grasp Network (PGN) for generating grasps at each image pixel. In the following, we describe in detail the various modules of DSGD.

Base Network

The purpose of the base network is to act as a feature extractor. We extract features from the intermediate layers of a CNN such as DenseNets (Huang et al. 2017), and use the features to learn grasp representations at different hierarchical levels. The basic building block of DenseNets (Huang et al. 2017) is a *Dense block*: bottleneck convolutions interconnected through dense connections. Specifically, a dense block consists of N_l number of layers termed *Dense Layers* which share information from all the preceding layers connected to the current layer through skip connections (Huang et al. 2017). Fig. 3 shows the structure of a dense block with $N_l = 6$ dense layers. Each dense layer consists of 1×1 and 3×3 convolutions followed by Batch Normalization (Ioffe and Szegedy 2015) and a Rectified Linear Unit (ReLU). The output of the l^{th} dense layer ($\mathcal{X}_l$) in a dense block can be written as:

$$\mathcal{X}_l = [\mathcal{X}_0, ..., \mathcal{X}_{l-1}], \tag{4}$$

where $[\cdots]$ represents concatenation of the features produced by the layers $0, ..., l-1$.

Global Grasp Network (GGN)

Our GGN structure is composed of two sub-networks as shown in Fig. 1-A: a Global Grasp Prediction Network (GGPN) for generating grasp pose ($[x_g, y_g, w_g, h_g, \theta_g]$) and a Grasp Evaluation Network (GEN) for predicting grasp confidence (ρ_g). The GGPN structure is composed of a dense block, an averaging operation, a

[1] $s = 1$ for $\boldsymbol{M}_{xy}$, $\boldsymbol{M}_w$, and $\boldsymbol{M}_h$, and $s = N_\theta$ for $\boldsymbol{M}_\theta$.

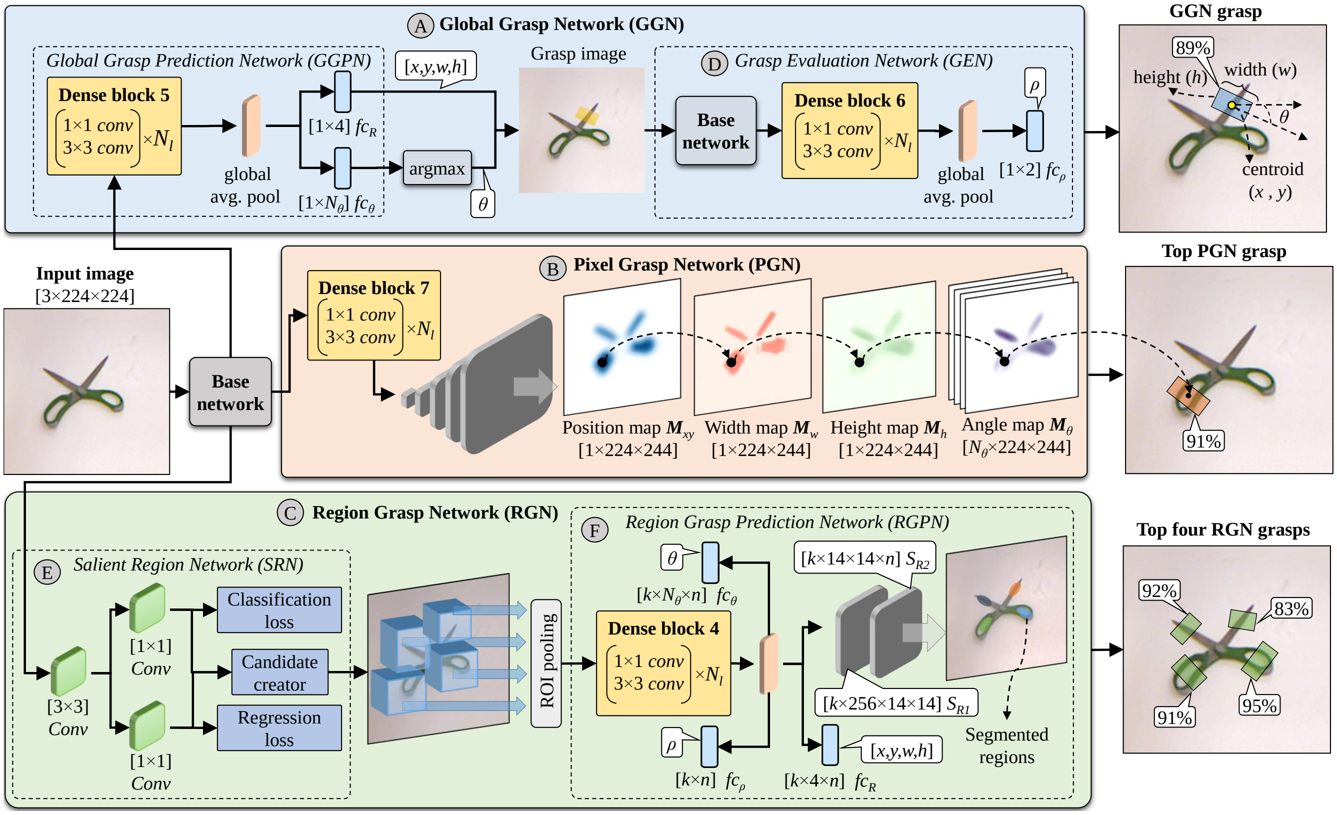

Figure 1: Overview of our DSGD architecture. Given an image as input, DSGD uses a base network to extract features which are fed into a Global Grasp Network (A), a Pixel Grasp Network (B), and a Region Grasp network (C), to produce grasp candidates. The global model produces a single grasp per image and uses an independent Grasp Evaluation Network (D) to produce grasp confidence. The pixel-level model uses a fully convolutional network and produces grasps at every pixel. The region-level model uses a Salient Region Network (E) to generate region proposals which are fed into a Region Grasp Prediction Network (F) to produce salient region segmentations and their corresponding grasp poses. During inference, DSGD switches between the GGN, the PGN, and the RGN models based on their confidence scores.

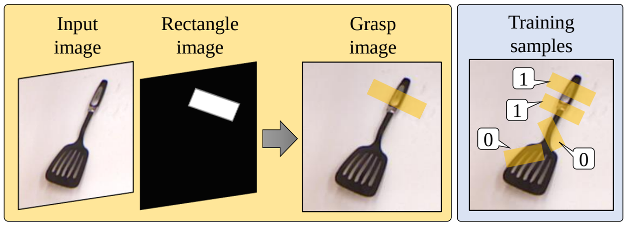

Figure 2: Left: A grasp image is generated by replacing the blue channel of the input image with a binary rectangle image produced from a grasp pose. Right: Our Grasp Evaluation Network is trained using grasp images labelled in terms of valid (1) and invalid (0) grasp rectangles.

4−dimensional fully connected layer for predicting the parameters $[x_g, y_g, w_g, h_g]$, and a N_θ−dimensional fully connected layer for predicting θ_g. The GEN structure is similar to GGPN except that GEN has a single 2−dimensional fully connected layer for predicting ρ_g. The input to GEN is a grasp image which is produced by replacing the *Blue* channel of the input image with a binary rectangle image generated from the output of GGPN as shown in Fig. 2. Let $R_{g_i} = [x_{g_i}, y_{g_i}, w_{g_i}, h_{g_i}]$, θ_{g_i} and ρ_{g_i} denote the predicted values of a global grasp for the i^{th} image. We define the loss of the GGPN and the GEN models over $\boldsymbol{K}$ images as:

$$L_{ggpn} = \sum_{i\in\boldsymbol{K}} \left((1-\lambda_1)L_{reg}(R_{g_i}, R^*_{g_i}) + \lambda_1 L_{cls}(\theta_{g_i}, \theta^*_{g_i})\right), \tag{5}$$

$$L_{gen} = \sum_{i\in\boldsymbol{K}} L_{cls}(\rho_{g_i}, \rho^*_{g_i}), \tag{6}$$

where $R^*_{g_i}$, $\theta^*_{g_i}$, and $\rho^*_{g_i}$ represent the ground-truths. The term L_{reg} is a regression loss defined as:

$$L_{reg}(R, R^*) = ||R - R^*||/||R^*||_2. \tag{7}$$

The term L_{cls} is a classification loss defined as:

$$L_{cls}(\mathrm{x}, c) = -\sum_{c=1}^{N_\theta} \mathcal{Y}_{\mathrm{x},c} \log(p_{\mathrm{x},c}), \tag{8}$$

where $\mathcal{Y}$ is a binary indicator if class label c is the correct classification for observation x, and p is the predicted probability of observation x of class c.

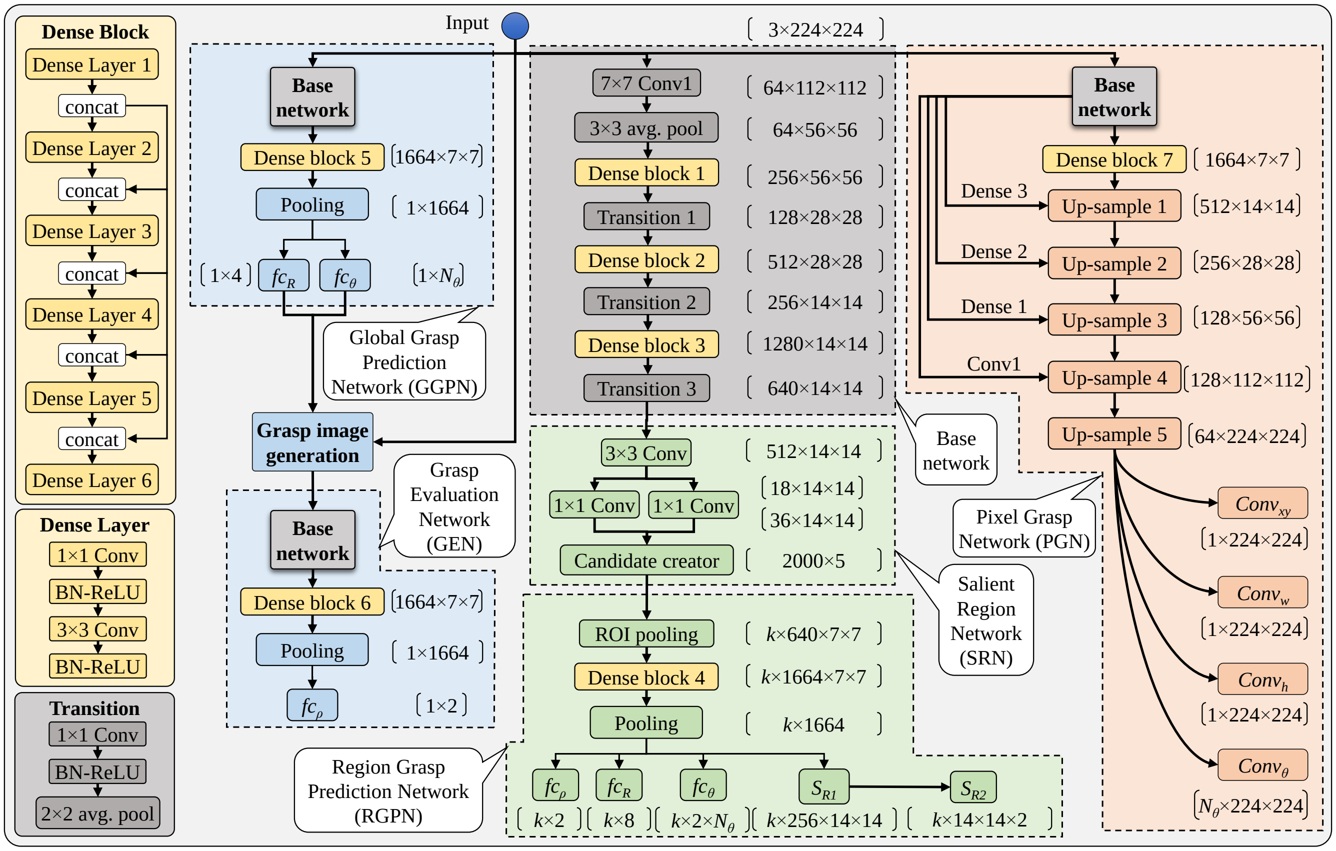

Figure 3: Detailed architecture of our DSGD with a DenseNet (Huang et al. 2017) as its base network.

Region Grasp Network (RGN)

The RGN structure is composed of two sub-networks as shown in Fig. 1-C: a Salient Region Network (SRN) for producing salient region proposals, and a region grasp prediction network (RGPN) for producing salient region segmentations and their corresponding grasp poses.

Salient Region Network (SRN): Here, we use the features extracted from the base network to generate proposals defined by the location (x_{sr}, y_{sr}), width (w_{sr}), height (h_{sr}), and confidence (ρ_{sr}) of non-oriented rectangles which encompass salient parts of the image (e.g., handles, extrusions). For this, we first generate a fixed number of rectangles using the Region of Interest (ROI) method of (He et al. 2017). Next, we use the features from the base network and optimize a Mean Squared Loss on the rectangle coordinates and a Cross Entropy Loss on the rectangle confidence scores. Let $T_i = [x_{sr}, y_{sr}, w_{sr}, h_{sr}]$ denote the parameters of the i^{th} predicted rectangle, and ρ_{sr_i} denote its probability whether it belongs to a graspable region or a non-graspable region. The loss of SRN over $\boldsymbol{I}$ proposals is given by:

$$L_{srn} = \sum_{i\in \boldsymbol{I}} \left((1-\lambda_2)L_{reg}(T_i, T_i^*) + \lambda_2 L_{cls}(\rho_{sr_i}, \rho^*_{sr_i})\right), \tag{9}$$

where $\rho^*_{sr_i} = 0$ for a non-graspable region and $\rho^*_{sr_i} = 1$ for a graspable region. The term T_i^* represents the ground truth candidate corresponding to $\rho^*_{sr_i}$.

Region Grasp Prediction Network (RGPN): Here, we produce salient region segmentations and their corresponding grasp poses using the proposals predicted by SRN ($k = 50$ in our implementation). For this, we crop features from the output feature maps of the base network using the Region of Interest (ROI) pooling method of (He et al. 2017). The cropped features are then fed to *Dense block 4* which produces feature maps of $k \times 1664 \times 7 \times 7-$dimensions as shown in Fig. 3. These feature maps are then squeezed to $k \times 1664-$dimensions through a global average pooling, and fed to a segmentation branch (with two up-sampling layers $S_{R1} \in \mathbb{R}^{k\times 256\times 14\times 14}$ and $S_{R2} \in \mathbb{R}^{k\times 14\times 14\times n}$), which produces a segmentation mask for each salient region as shown in Fig. 1-F. The squeezed features are also fed to three fully connected layers $fc_R \in \mathbb{R}^{k\times 4\times n}$, $fc_\theta \in \mathbb{R}^{k\times N_\theta \times n}$, and $fc_\rho \in \mathbb{R}^{k\times 2}$ which produce class-specific grasps for the segmented regions. Let $R_{r_i} = [x_{r_i}, y_{r_i}, w_{r_i}, h_{r_i}]$, θ_{r_i}, and ρ_{r_i} denote the predicted values of a region-level grasp for the i^{th} salient region, and $\mathcal{S}_i \in \mathbb{R}^{14\times 14\times n}$ denotes the corresponding predicted segmentation. The loss of the RGPN model is defined over $\boldsymbol{I}$ salient regions as:

$$L_{rgpn} = \sum_{i \in \boldsymbol{I}} (L_{reg}(R_{r_i}, R^*_{r_i}) + \lambda_3 L_{cls}(\theta_{r_i}, \theta^*_{r_i}) + \lambda_3 L_{cls}(\rho_{r_i}, \rho^*_{r_i}) + L_{seg}(\mathcal{S}_i, \mathcal{S}^*_i)), \tag{10}$$

where R^*, θ^*, ρ^*, and $\mathcal{S}^*$ represent the ground truths. The term $\rho^*_{r_i} = 0$ for a non-graspable region and $\rho^*_{r_i} = 1$ for a graspable region. The term L_{seg} represents a pixel-wise binary cross-entropy loss used to learn segmentations of salient regions. It is given by:

$$L_{seg} = -\frac{1}{|\mathcal{S}_i|} \sum_{j \in \mathcal{S}_i} (y_j \log(\hat{y}_j) + (1-y_j)\log(1-\hat{y}_j)), \tag{11}$$

where, y_j represents the ground truth value and $\hat{y}_j$ denotes the predicted value for a pixel $j \in \mathcal{S}_i$. Learning segmentation based grasp poses enables the model to produce grasp confidence maps where the confidence scores follow Gaussian distributions with the highest confidence at the center of the segmented region. This produces region-centred grasps and therefore better localization results. The total loss of our RGN model is given by:

$$L_{rgn} = L_{srn} + L_{rgpn}. \tag{12}$$

The terms λ_1, λ_2, and λ_3 in Eq. 5, Eq. 9, and Eq. 10 control the relative influence of classification over regression on the combined objective functions[2].

Pixel Grasp network (PGN)

Here, we feed the features extracted from the base network into *Dense block 7* followed by a group of upsampling layers which increase the spatial resolution of the features and produce feature maps of the size of the input image. These feature maps encode the parameters of the grasp pose at every pixel of the image. Let $\boldsymbol{M}_{xy_i}$, $\boldsymbol{M}_{w_i}$, $\boldsymbol{M}_{h_i}$, and $\boldsymbol{M}_{\theta_i}$ denote the predicted feature maps of the i^{th} image, respectively. We define the loss of the PGN model over $\boldsymbol{K}$ images as:

$$L_{pgn} = \sum_{i \in \boldsymbol{K}} (L_{reg}(\boldsymbol{M}_{xy_i}, \boldsymbol{M}^*_{xy_i}) + L_{reg}(\boldsymbol{M}_{w_i}, \boldsymbol{M}^*_{w_i}) + L_{reg}(\boldsymbol{M}_{h_i}, \boldsymbol{M}^*_{h_i}) + L_{cls}(\boldsymbol{M}_{\theta_i}, \boldsymbol{M}^*_{\theta_i})), \tag{13}$$

where $\boldsymbol{M}^*_{xy_i}$, $\boldsymbol{M}^*_{w_i}$, $\boldsymbol{M}^*_{h_i}$, and $\boldsymbol{M}^*_{\theta_i}$ represent the ground-truths.

Training and Implementation

For the global model, we trained the GGPN and the GEN sub-networks independently. For the region-based and the pixel-based models, we trained the networks in an end-to-end manner. Specifically, we initialized the weights of the base network with the weights pre-trained on ImageNet. For the *Dense blocks (4-7)*, the fully connected layers of GGPN, GEN, SRN, and RGPN, and the fully convolutional layers of PGN, we initialized the weights from zero-mean Gaussian distributions (standard deviation set to 0.01, biases set to 0), and trained the networks using the loss functions in Eq. 5, Eq. 6, Eq. 9, Eq. 12, and Eq. 13, respectively for 150 epochs. The starting learning rate was set to 0.01 and divided by 10 at 50% and 75% of the total number of epochs. The parameter decay was set to 0.0005 on the weights and biases. Our implementation is based on the framework of Torch library (Paszke et al. 2017). Training was performed using ADAM optimizer and data parallelism on four Nvidia Tesla K80 GPU devices. For grasp selection during inference, DSGD selects the most confident region-level grasp if its confidence score is greater than a confidence threshold (δ_{rgn}), otherwise DSGD switches to the PGN branch and selects the most confident pixel-level grasp. If the most confident pixel-level grasp has a confidence score less than δ_{pgn}, DSGD switches to the GGN branch and selects the global grasp as the output. Experimentally, we found that $\delta_{rgn} = 0.95$ and $\delta_{pgn} = 0.90$ produced the best grasp detection results.

Experiments

We evaluated DSGD for grasp detection on the popular Cornell grasp dataset (Lenz, Lee, and Saxena 2015), which contains 885 RGB-D images of 240 objects. The ground-truth is available in the form of grasp-rectangles. We also evaluated DSGD for multi-object grasp detection in new environments. For this, we used the multi-object dataset of (Asif, Tang, and Harrer 2018b) which consists of 6896 RGB-D images of indoor scenes containing multiple objects placed in different locations and orientations. The dataset was generated using an extended version of the scene labeling framework of (Asif, Bennamoun, and Sohel 2017a) and (Asif, Bennamoun, and Sohel 2016). For evaluation, we used the object-wise splitting criteria (Lenz, Lee, and Saxena 2015) for both the Cornell grasp dataset and our multi-object dataset. The object-wise splitting splits the object instances randomly into train and test subsets (i.e., the training set and the test set do not share any images from the same object). This strategy evaluates how well the model generalizes to unseen objects. For comparison purposes, we followed the procedure of (Redmon and Angelova 2015) and substituted the blue channel with the depth image, where the depth values are normalized between 0 and 255. We also performed data augmentation through random rotations. For grasp evaluation, we used the "rectangle-metric" proposed in (Jiang, Moseson, and Saxena 2011). A grasp is considered 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 higher than 25%. The Jaccard index for a predicted rectangle $\boldsymbol{\mathcal{R}}$ and a ground-truth rectangle $\boldsymbol{\mathcal{R}}^*$ is defined as:

$$J(\boldsymbol{\mathcal{R}}^*, \boldsymbol{\mathcal{R}}) = \frac{|\boldsymbol{\mathcal{R}}^* \cap \boldsymbol{\mathcal{R}}|}{|\boldsymbol{\mathcal{R}}^* \cup \boldsymbol{\mathcal{R}}|}. \tag{14}$$

Single-Object Grasp Detection

Table 1 shows that our DSGD achieved the best grasp detection accuracy on the Cornell grasp dataset compared to the other methods. We attribute this improvement to two main reasons: **First**, the proposed hierarchical grasp generation enables DSGD to produce grasps and their confi-

[2] For experiments, we set the parameters λ_1, λ_2, and λ_3 to 0.4.

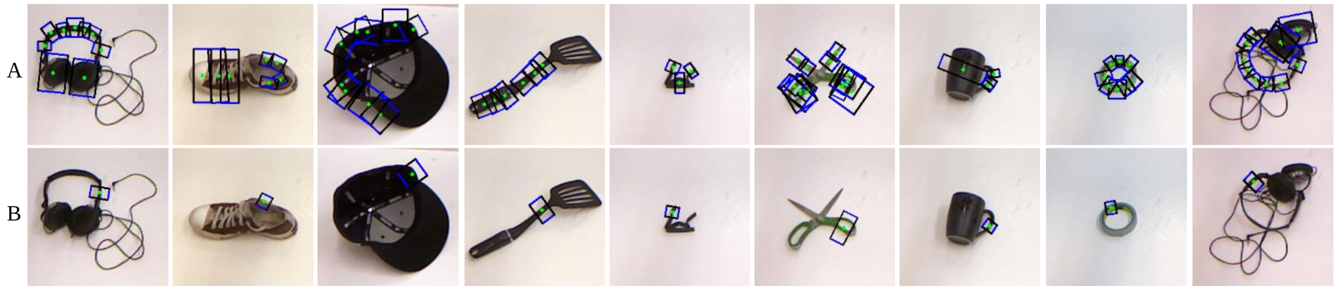

Figure 4: Grasp detection results of our DSGD (B) on some challenging objects of the Cornell grasp dataset. Ground truths are shown in A.

Table 1: Grasp evaluation on the Cornell grasp dataset in terms of average grasp detection accuracy.

Method	Accuracy (%)
(Jiang *et. al.* 2011) Fast search	58.3
(Lenz *et. al.* 2015) Deep learning	75.6
(Redmon *et. al.* 2015) MultiGrasp	87.1
(Kumra *et. al.* 2017) ResNets	88.9
(Guo *et. al.* 2017) Hybrid-Net	89.1
(Asif *et. al.* 2018) GraspNet	90.2
(Chu *et. al.* 2018) Multi-grasp	96.1
(this work) DSGD	**97.7**

Table 2: Comparison of the individual networks of the proposed DSGD in terms of grasp accuracy (%) on the Cornell grasp dataset.

Base network	Global model	Local models		DSGD
	GGN	PGN	RGN	
ResNet50	86.8	94.1	96.3	**96.7**
DenseNet	88.9	95.4	97.0	**97.7**

dence scores from both global and local contexts. This enables DSGD to effectively recover from the errors of the global (Kumra and Kanan 2017) or local methods (Guo et al. 2017). **Second**, the use of dense feature fusion enables the networks to learn more discriminative features compared to the models used in (Kumra and Kanan 2017; Guo et al. 2017). Fig. 4 shows grasps produced by our DSGD on some images of the Cornell grasp dataset.

Significance of Combining Global and Local Models: Table 2 shows a quantitative comparison of the individual models of our DSGD in terms of grasp accuracy on the Cornell grasp dataset, for different CNN structures as the base network. The base networks we tested include: ResNets (He et al. 2016) and DenseNets (Huang et al. 2017). Table 2 shows that on average the local models (PGN and RGN) produced higher grasp accuracy compared to the global model (GGN). The global and the local models have their own pros and cons. The global model uses the entire image information and learns an average of the ground-truth grasps. Although, the global-grasps are accurate for most of the objects, the grasps tend to lie in the middle of circular symmetric objects resulting in localization errors as highlighted in red in Fig. 5-B. The PGN model on the other hand operates at the pixel-level and produces correct grasp localizations for these challenging objects as shown in Fig. 5-C. However, pixel-based model is susceptible to outliers in the position prediction maps which result in localization errors as highlighted in red in Fig. 5-D. Our RGN model works at a semi-global level while maintaining large receptive fields. It predicts grasps using segmentation information of salient parts of the image which highly likely encode graspable parts of an object as shown in Fig. 5-F. Consequently, RGN is less susceptible to pixel-level outliers (see Fig 6-B and Fig 6-C) and does not suffer global averaging errors as shown in Fig. 5-G. Furthermore, the grasp confidences produced by the RGN model follow Gaussian distributions where the confidence scores are the highest at the center of the segmented salient regions (see Fig. 6-E). This results in grasps which are more stable compared to the PGN model, where the predicted grasp confidences follow uniform distributions along the surfaces of the objects resulting in several unstable grasps as shown in Fig. 6-D. Our DSGD takes advantage of both the global context and local predictions and produces highly accurate grasps as shown in Fig. 5-H.

Ablative Study of the Proposed DSGD: The growth rate parameter $\mathcal{W}$ refers to the number of output feature maps of each dense layer and therefore controls the depth of the network. Table 3 shows that a large growth rate and wider dense blocks (i.e., more number of layers in the dense blocks) increase the average accuracy from 96.9% to 97.7% at the expense of low runtime speed due to the overhead from additional channels. Table 3 also shows that a *lite* version of our detector (DSGD-*lite*) can run at 6*fps* making it suitable for real-time applications.

Multi-Object Grasp Detection

Table 4 shows our grasp evaluation on the multi-object dataset. The results show that on average, our DSGD improves grasp detection accuracy by 9% and 2.4% compared to the pixel-level and region-level models, respectively. Fig. 7 shows qualitative results on the images of our multi-object dataset. The results show that our DSGD successfully generates correct grasps for multiple objects in real-world scenes containing background clutter. The generalization capability of our model is attributed to the proposed hierarchical image-to-grasp mappings, where the proposed region-level

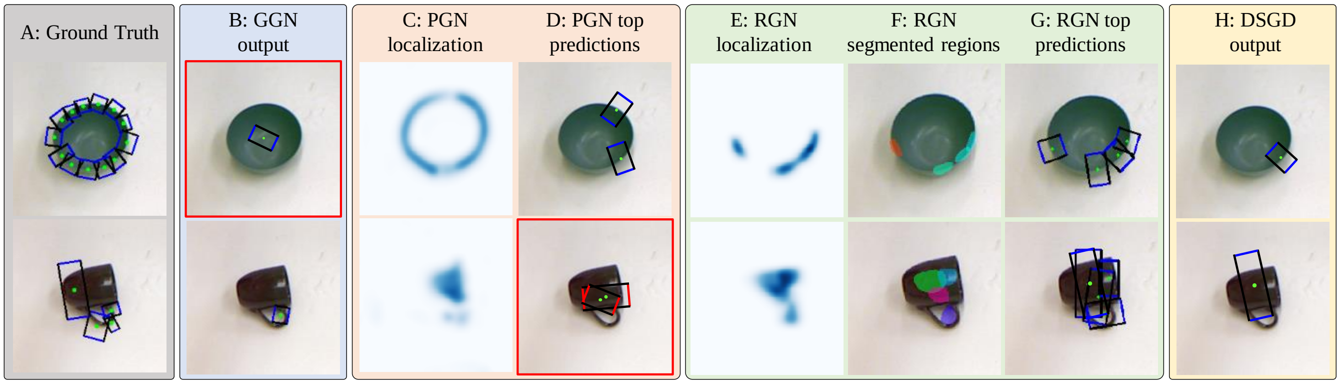

Figure 5: Qualitative comparison of grasps produced by the proposed GGN, PGN, RGN, and DSGD models. The results show that our DSGD effectively recovers from the errors of the individual models. Incorrect predictions are highlighted in red.

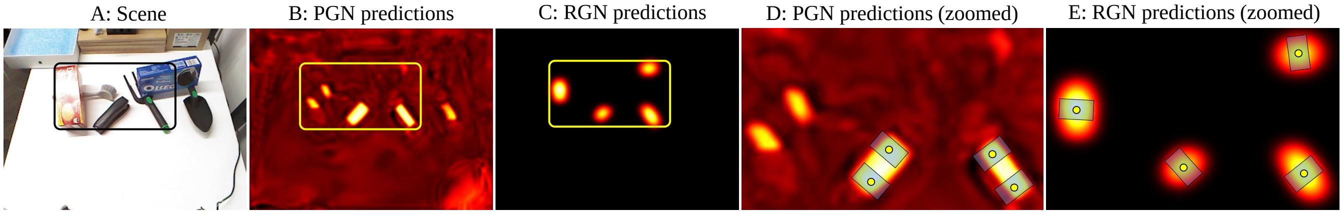

Figure 6: Comparison of PGN and RGN predictions. The intensity of the prediction maps represent grasp confidences. The RGN predictions (C) are less prone to pixel-level outliers (e.g., due to noise) compared to the PGN predictions (B). Furthermore, the RGN predictions follow Gaussian distributions where the confidence values are the highest at the center of the segmented salient regions (E) producing more stable grasps compared to the PGN predictions where confidence values follow uniform distributions along the surfaces of the objects (D).

Table 3: Ablation study of our DSGD (with DenseNet as the base network) on the Cornell grasp dataset in terms of the growth rate ($\mathcal{W}$) and the number of dense layers (N_l) of the GGN, PGN, and RGN sub-networks.

Model	RGN					PGN					GGN					Accuracy	Speed
	$\mathcal{W}$	N_{l_1}	N_{l_2}	N_{l_3}	N_{l_5}	$\mathcal{W}$	N_{l_1}	N_{l_2}	N_{l_3}	N_{l_7}	$\mathcal{W}$	N_{l_1}	N_{l_2}	N_{l_3}	N_{l_4}	(%)	(fps)
DSGD-*lite*	32	6	12	24	16	32	6	12	24	16	32	6	12	24	16	96.9	**6**
DSGD-A	32	6	12	32	32	32	6	12	32	32	32	6	12	24	16	97.1	5
DSGD-B	48	6	12	36	24	32	6	12	32	32	32	6	12	24	16	97.3	4
DSGD-C	48	6	12	36	24	32	6	12	48	32	32	6	12	24	16	97.5	4
DSGD-D	48	6	12	36	24	32	6	12	24	16	32	6	12	24	16	**97.7**	4
DSGD-E	48	6	12	36	24	48	6	12	36	24	32	6	12	24	16	97.4	3

network and the proposed pixel-level network learn to associate grasp poses to salient regions and salient pixels in the image data, respectively. These salient regions and pixels encode object graspable parts (e.g., boundaries, corners, handles, extrusions) which are generic (i.e., have similar appearance and structural characteristics) across a large variety of objects generally found in indoor environments. Consequently, the proposed hierarchical mappings learned by our models successfully generalize to new object instances during testing. This justifies the practicality of our DSGD for real-world robotic grasping.

Table 4: Grasp evaluation on our multi-object dataset.

Base network	Grasp accuracy			Robotic grasp
	PGN	RGN	DSGD	success
ResNet	86.5%	93.4%	**95.8%**	89%
DenseNet	87.4%	94.7%	**97.2%**	90%

Robotic Grasping

Our robotic grasping setup consists of a Kinect for image acquisition and a 7 degrees of freedom robotic arm which is tasked to grasp, lift, and take-away the objects placed within the robot workspace. For each image, DSGD generates multiple grasp candidates as shown in Fig. 8. For grasp execution, we select a random candidate which is located within the robot workspace and has confidence greater than 90%. A robotic grasp is considered successful if the robot grasps the target object (verified through force sensing in the grip-

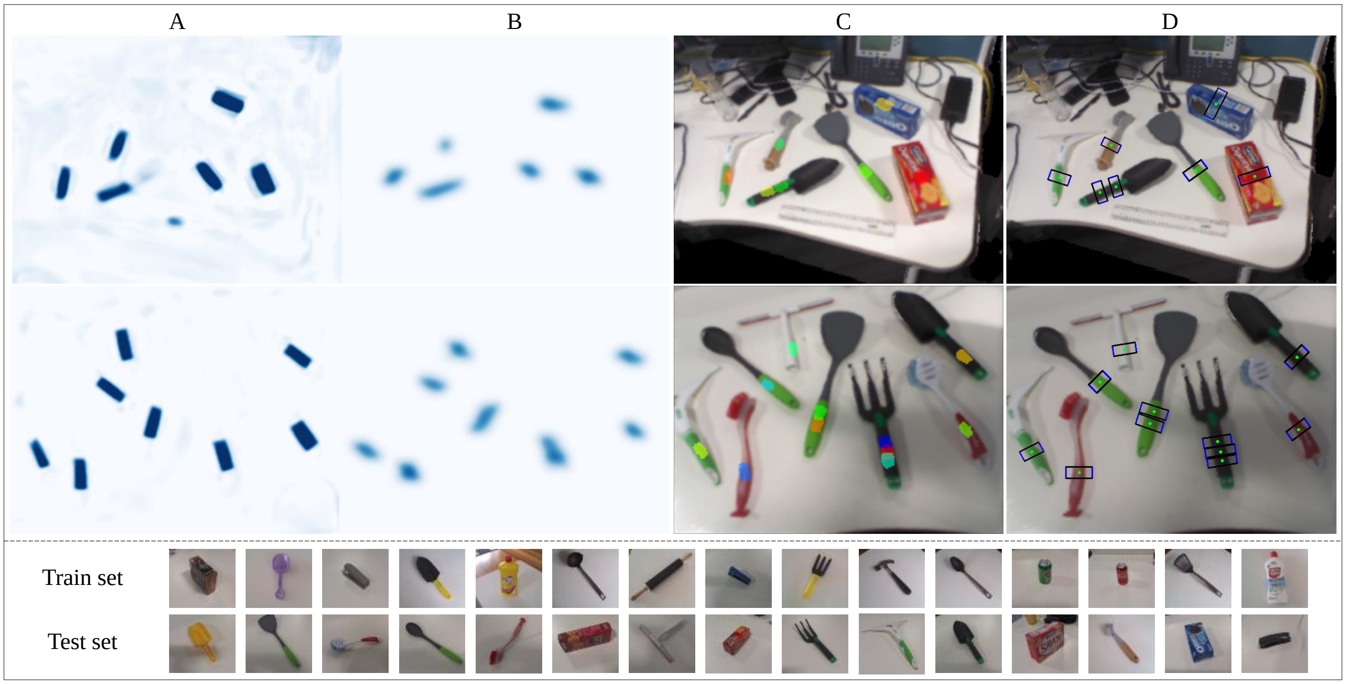

Figure 7: Grasp evaluation on our multi-object dataset. The localization outputs of our pixel-level (PGN) and region-level (RGN) grasp models are shown in (A) and (B), respectively. The intensity of the prediction maps A and B represent grasp confidences. The segmented regions produced by our RGN model are shown in (C). Note that we only show grasps with confidence scores higher than 90% in (D). Our grasp detection results on Kinect video streams are available at: https://youtu.be/tA2qgtbTT98

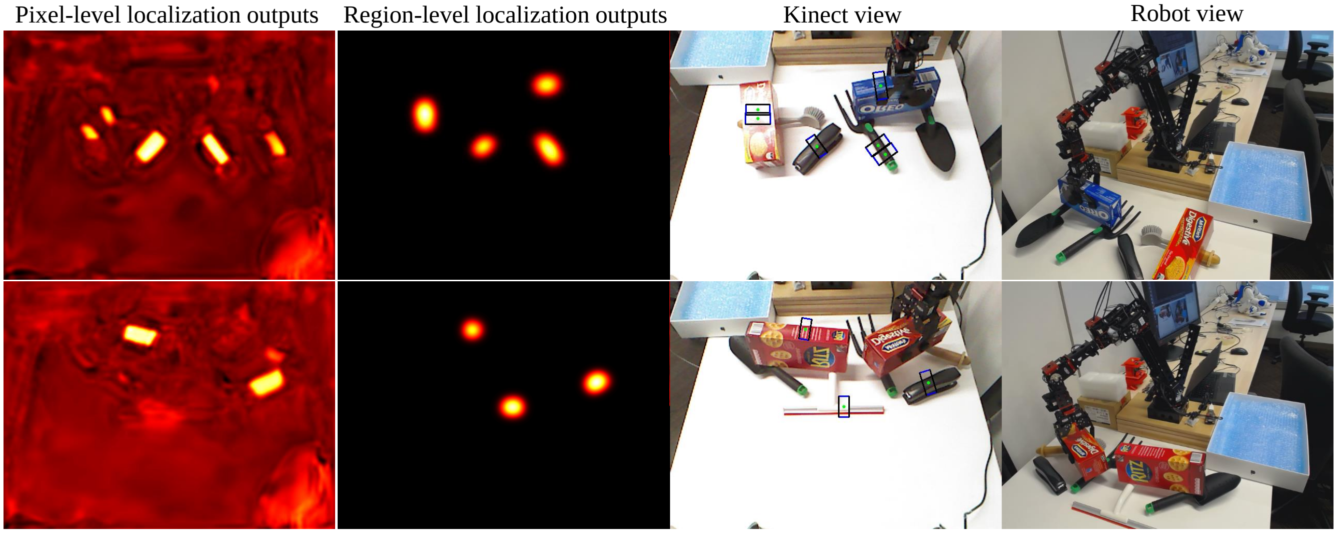
Pixel-level localization outputs
Region-level localization outputs
Kinect view
Robot view

Figure 8: Experimental setting for real-world robotic grasping. See video of our experiments at: https://youtu.be/Bn_dQ8vcNzs

per), holds it in air for 3 seconds and takes it away from the robot workspace. The objects are placed in random positions and orientations to remove bias related to the object pose. Table 4 shows the success rates computed over 200 grasping trials. The results show that we achieved grasp success rates of 90% with DenseNet as the base network. Some failure cases include objects with non-planar grasping surfaces (e.g., brush). However, this can be improved by multi-finger grasps. We leave this for future work as our robotic arm only supports parallel grasps.

Conclusion and Future Work

We presented *Densely Supervised Grasp Detector* (DSGD), which generates grasps and their confidence scores at different image hierarchical levels (i.e., global-, region-, and pixel-levels). Experiments show that our proposed hierarchical grasp generation produces superior grasp accuracy compared to the state-of-the-art on the Cornell grasp dataset. Our evaluations on videos from Kinect and robotic grasping experiments show the capability of our DSGD for producing stable grasps for unseen objects in new environments. In future, we plan to reduce the computational burden of our DSGD through parameter-pruning for low-powered GPU devices.

References

Asif, U.; Bennamoun, M.; and Sohel, F. 2016. Simultaneous dense scene reconstruction and object labeling. In *ICRA*, 2255–2262. IEEE.

Asif, U.; Bennamoun, M.; and Sohel, F. 2017a. A multimodal, discriminative and spatially invariant cnn for rgb-d object labeling. *IEEE Transactions on Pattern Analysis and Machine Intelligence*.

Asif, U.; Bennamoun, M.; and Sohel, F. A. 2017b. Rgb-d object recognition and grasp detection using hierarchical cascaded forests. *IEEE Transactions on Robotics*.

Asif, U.; Tang, J.; and Harrer, S. 2018a. Ensemblenet: Improving grasp detection using an ensemble of convolutional neural networks. In *British Machine Vision Conference (BMVC)*.

Asif, U.; Tang, J.; and Harrer, S. 2018b. 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*, 4875–4882.

Chu, F.-J.; Xu, R.; and Vela, P. A. 2018. Real-world multi-object, multigrasp detection. *IEEE Robotics and Automation Letters* 3(4):3355–3362.

Guo, D.; Sun, F.; Liu, H.; Kong, T.; Fang, B.; and Xi, N. 2017. A hybrid deep architecture for robotic grasp detection. In *ICRA*, 1609–1614. IEEE.

He, K.; Zhang, X.; Ren, S.; and Sun, J. 2016. Deep residual learning for image recognition. In *CVPR*, 770–778.

He, K.; Gkioxari, G.; Dollár, P.; and Girshick, R. 2017. Mask r-cnn. In *ICCV*, 2980–2988. IEEE.

Huang, G.; Liu, Z.; Weinberger, K. Q.; and van der Maaten, L. 2017. Densely connected convolutional networks. In *CVPR*, volume 1, 3.

Ioffe, S., and Szegedy, C. 2015. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *ICML*, 448–456.

Jiang, Y.; Moseson, S.; and Saxena, A. 2011. Efficient grasping from rgbd images: Learning using a new rectangle representation. In *ICRA*, 3304–3311. IEEE.

Johns, E.; Leutenegger, S.; and Davison, A. J. 2016. Deep learning a grasp function for grasping under gripper pose uncertainty. In *IROS*, 4461–4468. IEEE.

Kumra, S., and Kanan, C. 2017. Robotic grasp detection using deep convolutional neural networks. In *IROS*. IEEE.

Lenz, I.; Lee, H.; and Saxena, A. 2015. Deep learning for detecting robotic grasps. *IJRR* 34(4-5):705–724.

Levine, S.; Pastor, P.; Krizhevsky, A.; Ibarz, J.; and Quillen, D. 2016. Learning hand-eye coordination for robotic grasping with deep learning and large-scale data collection. *IJRR* 0278364917710318.

Mahler, J.; Pokorny, F. T.; Hou, B.; Roderick, M.; Laskey, M.; Aubry, M.; Kohlhoff, K.; Kröger, T.; Kuffner, J.; and Goldberg, K. 2016. 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*, 1957–1964. IEEE.

Mahler, J.; Liang, J.; Niyaz, S.; Laskey, M.; Doan, R.; Liu, X.; Ojea, J. A.; and Goldberg, K. 2017. Dex-net 2.0: Deep learning to plan robust grasps with synthetic point clouds and analytic grasp metrics. *Robotics: Science and Systems (RSS)*.

Morrison, D.; Corke, P.; and Leitner, J. 2018. Closing the loop for robotic grasping: A real-time, generative grasp synthesis approach. *arXiv preprint arXiv:1804.05172*.

Myers, A.; Teo, C. L.; Fermüller, C.; and Aloimonos, Y. 2015. Affordance detection of tool parts from geometric features. In *ICRA*, 1374–1381.

Paszke, A.; Gross, S.; Chintala, S.; Chanan, G.; Yang, E.; DeVito, Z.; Lin, Z.; Desmaison, A.; Antiga, L.; and Lerer, A. 2017. Automatic differentiation in pytorch.

Pinto, L., and Gupta, A. 2016. Supersizing self-supervision: Learning to grasp from 50k tries and 700 robot hours. In *ICRA*, 3406–3413. IEEE.

Redmon, J., and Angelova, A. 2015. Real-time grasp detection using convolutional neural networks. In *ICRA*, 1316–1322. IEEE.

Ren, S.; He, K.; Girshick, R.; and Sun, J. 2015. Faster r-cnn: Towards real-time object detection with region proposal networks. In *NIPS*, 91–99.

Saxena, A.; Driemeyer, J.; and Ng, A. Y. 2008. Robotic grasping of novel objects using vision. *IJRR* 27(2):157–173.

Varley, J.; Weisz, J.; Weiss, J.; and Allen, P. 2015. Generating multi-fingered robotic grasps via deep learning. In *IROS*, 4415–4420. IEEE.

Zeng, A.; Song, S.; Yu, K.-T.; Donlon, E.; Hogan, F. R.; Bauza, M.; Ma, D.; Taylor, O.; Liu, M.; Romo, E.; et al. 2017. Robotic pick-and-place of novel objects in clutter with multi-affordance grasping and cross-domain image matching. *arXiv preprint arXiv:1710.01330*.

Zeng, A.; Song, S.; Yu, K.-T.; Donlon, E.; Hogan, F. R.; Bauza, M.; Ma, D.; Taylor, O.; Liu, M.; Romo, E.; Fazeli, N.; Alet, F.; Dafle, N. C.; Holladay, R.; Morona, I.; Nair, P. Q.; Green, D.; Taylor, I.; Liu, W.; Funkhouser, T.; and Rodriguez, A. 2018. Robotic pick-and-place of novel objects in clutter with multi-affordance grasping and cross-domain image matching. In *ICRA*.