DENSELY CONNECTED DECONVOLUTIONAL NETWORK FOR SEMANTIC SEGMENTATION

Jun Fu[⋆†] *Jing Liu*[⋆] *Yuhang Wang*[⋆†] *Hanqing Lu*[⋆]

[⋆] National Laboratory of Pattern Recognition, Institute of
Automation, Chinese Academy of Sciences, Beijing, China;
[†] University of Chinese Academy of Sciences, Beijing, China

ABSTRACT

Recent progress in semantic segmentation has been driven by improving the spatial resolution under Fully Convolutional Networks(FCNs). To address this problem, we propose a Densely Connected Deconvolutional Network (DCDN) for semantic segmentation. In DCDN, multiple shallow deconvolutional networks, which are called as DCDN units, are stacked one by one to make the structure deeper and guarantee the fine recovery of localization information, meanwhile, the inter-unit and intra-unit dense connections are designed to make the network easy to train since the connections improve the flow of information and gradients throughout the network. Besides, the intermediate supervisions are applied to each DCDN unit to ensure the fast convergence. Extensive experiments on two urban scene datasets, i.e.,CamVid and GATECH, demonstrate that the proposed model achieves better performance than some state-of-the-art methods without using any post-processing, pretrained model, nor temporal information, whilst requiring less parameters.

Index Terms— Dense connection, Deconvolutional Network, Semantic Segmentation, Intermediate supervision

1. INTRODUCTION

Semantic segmentation is to predict the category of individual pixels in an image, and it has been one of the most important fields in computer vision. Recently, most of the semantic segmentation methods are based on the architecture of Full Convolutional Networks (FCNs) [1] which usually adopts a certain pretrained classification network and outputs a probability map per class for arbitrary-sized input. However, the classification network with necessary downsampling operations sacrifices the spatial resolution of feature maps to obtain the invariance to image transforms. The resolution reduction results in poor object delineation and small spurious regions in segmentation output.

Many approaches have been proposed to solve the above problem. One way is to apply dilated convolutions [2, 3] to enlarge the receptive field and capture larger context information without losing resolution. Another type of methods is to recover the spatial resolution by an upsampling or deconvolutional path [4, 5, 6, 7], which is also our focus in this paper. In those methods, the deconvolutional and unpooling layers are appended with a symmetric structure of the corresponding convolutional and pooling layers. However, the symmetric structure almost doubles the parameter size of the original convolutional structure. This brings much difficulty in training models. To make the model easy to convergence, Wang *et al.* [4] use VGG16 network [8] as pretrained weights to obtain better initial parameters of deconvolutional network, and Noh *et al.* [5] use two-stage training on single object images and multi-object images, respectively. The both works are based on the VGG framework, which is a relatively shallow structure in contrast to the lastest popular deeper models, e.g., ResNet [9] and DenseNet [10]. Consequently, the restricted learning ability of such a shallow structure is also an important problem that needs to be solved.

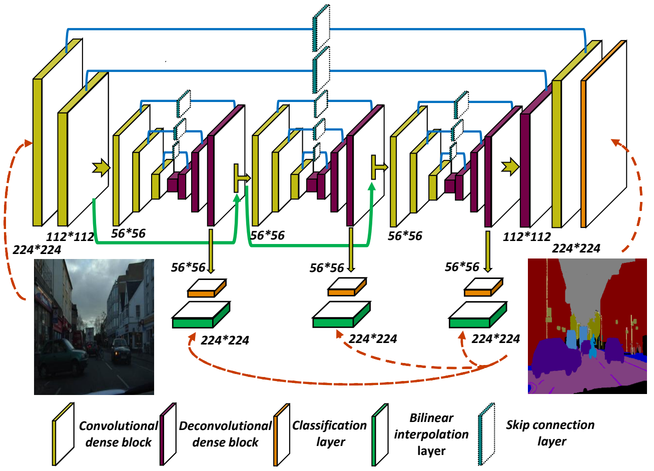

Fig. 1. Overall architecture of our approach. (Best viewed in color.)

To address the above problems, we design a deeper and easily optimized network called as Densely Connected Deconvolutional Network (DCDN) for semantic segmentation. In DCDN, some *shallow deconvolutional networks* (called as

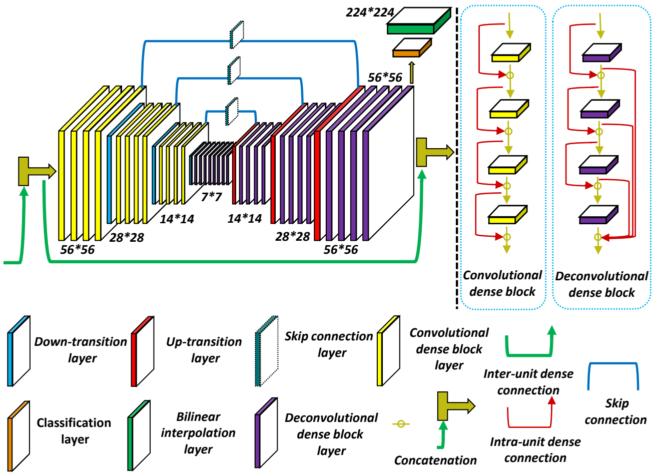
224*224
56*56
56*56
28*28
14*14
7*7
14*14
28*28
56*56
Convolutional dense block
Deconvolutional dense block
Down-transition layer
Up-transition layer
Skip connection layer
Convolutional dense block layer
Inter-unit dense connection
Skip connection
Classification layer
Bilinear interpolation layer
Deconvolutional dense block layer
Concatenation
Intra-unit dense connection

Fig. 2. The structure of DCDN unit. (Best viewed in color.)

DCDN units in our work) are stacked one by one, i.e., feeding the output of each unit as the input into its next unit. Compared to the deconvolutional models [4, 6, 7], such a stacked architecture increases the layers of the network by repeating DCDN units and shortens the path between downsampling and upsampling process with more shallow structure. In each DCDN unit, we adopt the similar architecture of DenseNets [10], but add an upsampling path to recover the full input resolution. As the number of the stacked DCDN units increases, the difficulty in training models becomes a major problem. We solve the problem from the following three sides. First, we apply the intermediate supervisions for DCDN units. Specifically, the outputs of each DCDN unit are mapped to pixel-wise labeling maps by a classification layer, and the result of the last unit is used to obtain the final prediction. Second, we import intra-unit and inter-unit dense connections to help the network optimization process. The dense connections as short paths from the early layers to the later layers are beneficial to the flow of information and gradients throughout the network. Specifically, the *intra-unit dense connections* are the direct links from the inputs of previous layers to the ones of back layers within a dense block, and the *inter-unit dense connections* are the short paths between any two adjacent D-CDN units, that is each unit inputs are the concatenation of the previous unit outputs and its inputs. Third, in a single DCD-N unit, *skip connections* are adopted to combine the feature maps from the downsampling path with the upsampling output, and a *skip connection layer* with convolution calculations is used for better feature fusion. The details of the proposed architecture are shown in **Fig. 1**. Our main contributions can be summarized as follows:

- DCDN stacks multiple shallow deconvolutional networks to enhance the learning ability of the network, and guarantee the fine recovery of localization information.
- The intermediate supervisions for DCDN units, the intra-unit and inter-unit dense connections, and the skip connections jointly result in faster convergence of the proposed network.
- The state-of-the-art performance on two challenging benchmarks is achieved, without any post-processing, pretrained model, nor any temporal information, whilst requiring less parameters.

2. MODEL

The detailed structure of our proposed model is described in this section. We first scan the basic architecture of DenseNets [10] which is the pioneer work to introduce dense connections. Then we combine dense connections into the deconvolutional framework to design a DCDN unit. Further on, we build our network by stacking DCDN units for more refined prediction, where inter-unit dense connections are employed. Some implementation details of our model are represented.

2.1. Review of DenseNets

Recently, DenseNets [10] have shown strong learning ability with less memory cost and efficient feature reuse, which achieve state-of-the-art performance on ImageNet [11]. Compared with ResNets, DenseNets make full use of short paths to avoid the vanishing gradient problem and improve the flow of information. DenseNets mainly consist of dense blocks and transition layers. The input of each convolutional layer within a dense block is the concatenation of all feature outputs of previous layers at a given resolution. Specifically, consider x_l is the output of the l^{th} layer in a dense block, x_l can be computed as follows:

$$x_l = H_l\left([x_0, x_1, \ldots, x_{l-1}]\right) \tag{1}$$

where $[x_0, x_1, \ldots, x_{l-1}]$ stands for the concatenation of the feature maps $x_0, x_1, \ldots, x_{l-1}$, and x_0 is the input of the dense block. Meanwhile, H_l is defined as a composite function of operations: Batch Normalization (BN)[12], a Rectified Linear Unit (ReLU), a 3 × 3 convolution operation (Conv) and dropout[13]. Each dense block is followed by a transition layer, which do convolution and pooling to change the number and the size of feature maps. Finally, a softmax classifier is attached to make prediction.

2.2. Shallow Deconvolutional Network (DCDN unit)

Inspired by the effectiveness of DenseNets [10], we import intra-unit dense connections and skip connections to design each DCDN unit, where a typical deconvolutional framework including both downsampling and upsampling operations is explored. It is noted that the dense connections in downsampling and upsampling paths are different in our design, and we will present the details in the following. The structure of DCDC unit is illustrated in **Fig. 2**.

In the downsampling path of DCDN unit, we adopt *convolutional dense blocks* and *down-transition layers* to encode features to a low resolution. In the block, the input of the convolutional layer (called as *convolutional dense block layer*) is the concatenation of the input and output of its previous convolutional layer. Each convolutional dense block is followed by a down-transition layer, which consists of a convolutional layer and a max-pooling layer.

Meanwhile, we apply *deconvolutional dense blocks* and *up-transition layers* to implement the upsampling operation. As illustrated in **Fig. 2**, the input of the convolutional layer (called as *deconvolutional dense block layer*) within a deconvolutional dense block is obtained in the same way as the convolutional dense block layer, while the block output is the concatenation of outputs of its convolutional layers. This dense connection pattern avoids the linear growth in the number of feature outputs of blocks, which reduces computational cost and memory demanding, meanwhile bring efficient feature reuse. Each deconvolutional dense block is followed by an up-transition layer which enlarges the size of feature maps through a deconvolutional operation.

Moreover, we use skip connection with a convolutional layer to connect the convolutional dense block layer with corresponding up-transition layer of the same resolution, which conveys spatial information from downsampling path to upsampling path. In order to obtain global perspective as much as possible, we stack more layers in the first deconvolutional dense block, which has lowest resolution in a DCDN unit.

2.3. Densely Connecting DCDN units

We extend on a single DCDN unit by stacking multiple DCDN units one-by-one with inter-unit dense connections. As illustrated in **Fig. 1**, the input of each DCDN unit is the concatenation of all previous unit outputs as well as the first unit input. For example, the input of the third unit is the concatenation of the output of the first two units, as well as the input of the first unit. The inter-unit dense connections make gradients spread easily from the later units to front units and make feature maps reused forwardly.

Meanwhile, intermediate supervision is also applied in our model to assist training. The output of each DCDN unit is fed to a classification layer with a bilinear interpolation layer attached to recover the spatial resolution of feature maps. In the cascade, the early predictions are coarse, while the subsequent DCDN units conduct refinements to the previous ones progressively.

As shown in **Fig. 1**, our highest resolution of DCDN unit is set to a quarter of input images. The reason for this design is that we can reduce GPU memory usage of a single DCDN unit to stack more units. Based on this design, the full network starts with a convolutional layer followed by convolutional dense blocks and down-transition layers to bring the resolution down to a quarter of input images. Then DCDN units are stacked to increase layers of the network. Finally, the feature maps from the last DCDN unit are fed to up-transition layers and dense blocks to recover the resolution of the feature maps to input images. Skip connection layers are also used for a finer information recovery. The pixel-wise cross-entropy loss is applied to all predictions in the network.

2.4. Implementation Details

In our design, all convolutional layers within a dense block are composed of BN, ReLU, and 3×3 Conv followed by dropout with probability 0.2. Down-transition layer is composed of BN, ReLU, 1×1 Conv, and dropout with probability 0.2, followed by 2×2 max pooling. Up-transition layer consists of a 4×4 deconvolution with stride 2. Skip convolution layer is composed of BN, ReLU, and 1×1 Conv. For the first deconvolutional dense block, we stack 8 convolution layers for better global perspective, while other blocks have 4 convolutional layers. The channel numbers of convolutional layers in a dense block are all set to 16.

We implement our network with Caffe [14], and optimize it using RMSprop [15], with batchsize of 20. We train our models in two stages. In the first stage, data augmented with random crops of 224×224 is used, and the initial learning rate is set to 0.001. In the second stage, we finetune our model with full images resized to 320×320, and learning rate of 1e-4. During inference, we also resize full images to 320×320 before fed into network. Meanwhile, we use the image and its mirror in both training and inference.

3. EXPERIMENTS

Extensive experiments are conducted on CamVid [16] dataset and CATECH [17] dataset. Our approach is compared with some of the state-of-the-art methods, and we achieve the best performance without any post-processing, pretrained model, nor any temporal information. Following [18], we apply **Global Avg** (Percentage of correctly labeled pixels over the whole annotated pixels) and **Mean IoU** (Percentage of correctly labeled pixels in a class over the union set of pixels predicted to this class and groundtruth, and then averaged over all classes) to evaluate our approach.

3.1. Results on CamVid Dataset

CamVid [16] is a street scene understanding dataset which consists of 5 video sequences. Following [6], we split the dataset into 367 training images, 100 validation images, and 233 test images. The resolution of each image is 360×480 and all images belong to 11 semantic categories. Results on the CamVid test set are reported in Table 1.

First, we verify the effect of stacking multiple DCDN units. As shown in Table 1, we refer to the network stacking k DCDN units as DCDN$_{Mk}$. We find that the performance

Table 1. Semantic segmentation results on CamVid

Method	Pretrained	Parameters	Mean IoU	Global Avg
SegNet [6]	Yes	29.5 M	55.6	88.5
Bayesian SegNet [7]	Yes	29.5 M	63.1	86.9
DeconvNet [5]	Yes	252 M	48.9	85.9
DeepLab-LFOV [2]	Yes	37.3 M	61.6	-
Dilation8 + FSO [19]	Yes	140.8 M	66.1	88.3
HDCNN-448+TL [20]	Yes	29.5 M	65.9	90.9
FC-DenseNet56 [18]	No	3.5 M	65.8	90.8
FC-DenseNet103 [18]	No	9.4 M	66.9	**91.5**
DCDN$_{M1}$	No	2.4 M	65.5	90.8
DCDN$_{M2}$	No	4.4 M	66.4	91.0
DCDN$_{M3}$	No	6.5 M	66.9	90.9
DCDN$_{M2+}$	No	12.8 M	66.8	91.3
DCDN$_{M4}$	No	8.7 M	**68.4**	91.4

increases with the growth of DCDN unit number. Particularly, the Mean IoU improves from 65.5 to 68.4 with the unit number increasing from 1 to 4. The noticeable trend indicates that, with increasing number of stacked units, the model benefits from deeper network as well as more parameters. Moreover, stacking multiple DCDN units makes a coarse-to-fine prediction process, thus improving the network performance. However, it also leads to more computational stress and harder optimization, so we stack up to 4 DCDN units as more units bring too slight improvements.

Second, we explore the effect of stacked network design. It is to compare the performance between the network stacking more DCDN units with shallow structure and the one stacking less DCDN units with deep structure at the same network depth. To this end, we design a new DCDN unit and refer to the corresponding network as DCDN$_{Mk+}$, where the number of convolutional layers in dense blocks are doubled. We compare a two-stacked network DCDN$_{M2+}$ and a four-stacked network DCDN$_{M4}$ which are of almost the same depth. As shown in Table 1, the performance of DCDN$_{M4}$ is 1.6 percent higher than DCDN$_{M2+}$. Moreover, the parameters of DCDN$_{M4}$ is much less than DCDN$_{M2+}$. The comparison indicates that stacking multiple shallow DCDN units is more effective and efficient than employing deep DCDN units at the same network depth, as shallow DCDN units with the stacked design are more conducive to the flow of information, optimization of the network.

When compared with other methods[2, 5, 6, 7, 19, 20, 18], our model DCDN$_{M4}$ achieves new state-of-the-art performance with less parameters. Among the methods, [19, 20] apply spatio-temporal information to boost their performance. And [6, 7, 2, 5, 19, 20] use the model pretrained on large-scale datasets to initialize the network which improves the performance remarkably. Besides, [2, 20] refine the outputs of the model by using CRF post-processing. It should be noticed that our model outperforms all the methods above, where any post-processing, pretrained model, and temporal information are all not applied to our model. While these processes are complementary to our approach and could bring additional improvements. Particularly, [18] also employs DenseNet based on a FCN-like [1] fashion for semantic segmentation. Our model DCDN$_{M4}$ performs much better than their best model FC-DenseNet103, as our model benefits from the stacked deconvoluational structure as well as more appropriate dense connections and skip connections.

Table 2. Semantic segmentation results on GATECH

Method	Temporal Info	Global Avg	Mean IoU
3D-V2V-scratch [21]	Yes	66.7	-
3D-V2V-finetune [21]	Yes	76.0	-
FC-DenseNet103 [18]	No	79.4	-
HDCNN-448+TL [20]	Yes	82.1	48.2
DCDN$_{M4}$	No	**83.5**	**49.0**

3.2. Results on GATECH Dataset

In order to verify the generalization of our models, we evaluate our network on GATECH [17] dataset, which is much larger than CamVid. GATECH dataset is a large video set of outdoor scenes which consists of 63 videos with 12241 frames for training and 38 videos with 7071 frames for testing. The dataset is labeled with 8 semantic classes which are sky, ground, solid, porous, cars, humans, vertical mix,and main mix.

Our results on GATECH test set are reported in Table 2. We employ the model DCDN$_{M4}$ pretrained on CamVid and finetune it on GATECH. Our model outperforms all current state-of-the-art methods, which confirms the effectiveness of our approach. Specially, our model trained without using any temporal information performs better than the models [21, 20] which exploit spatio-temporal relationships between video frames.

4. CONCLUSION

We have presented Densely Connected Deconvolutional Network (DCDN), a novel deep network architecture for semantic segmentation. We stack multiple DCDN units to make network deeper and realize a coarse-to-fine learning process, meanwhile dense connections are adopted to promote network optimization. Those designs are inspired by a general observation that deeper networks have stronger learning ability, and dense connections improve the information flow and feature reuse. We achieve the state-of-the-art performance on two challenging benchmarks for urban scene understanding, i.e. CamVid [16] and CATECH [17], without any post-processing, pretrained model, nor any temporal information.

5. REFERENCES

[1] Jonathan Long, Evan Shelhamer, and Trevor Darrell, "Fully convolutional networks for semantic segmentation," in *CVPR*, 2015, pp. 3431–3440.

[2] Liang-Chieh Chen, George Papandreou, Iasonas Kokkinos, Kevin Murphy, and Alan L Yuille, "Semantic image segmentation with deep convolutional nets and fully connected crfs," in *ICLR*, 2015.

[3] Fisher Yu and Vladlen Koltun, "Multi-scale context aggregation by dilated convolutions," in *ICLR*, 2015.

[4] Yuhang Wang, Jing Liu, Yong Li, Junjie Yan, and Hanqing Lu, "Objectness-aware semantic segmentation," in *Proceedings of the 2016 ACM on Multimedia Conference*. ACM, 2016, pp. 307–311.

[5] Hyeonwoo Noh, Seunghoon Hong, and Bohyung Han, "Learning deconvolution network for semantic segmentation," in *CVPR*, 2015, pp. 1520–1528.

[6] Vijay Badrinarayanan, Alex Kendall, and Roberto Cipolla, "Segnet: A deep convolutional encoder-decoder architecture for image segmentation," *arXiv preprint arXiv:1511.00561*, 2015.

[7] Alex Kendall, Vijay Badrinarayanan, and Roberto Cipolla, "Bayesian segnet: Model uncertainty in deep convolutional encoder-decoder architectures for scene understanding," *arXiv preprint arXiv:1511.02680*, 2015.

[8] Karen Simonyan and Andrew Zisserman, "Very deep convolutional networks for large-scale image recognition," in *ICLR*, 2015.

[9] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun, "Deep residual learning for image recognition," in *CVPR*, 2016, pp. 770–778.

[10] Gao Huang, Zhuang Liu, Kilian Q Weinberger, and Laurens van der Maaten, "Densely connected convolutional networks," *arXiv preprint arXiv:1608.06993*, 2016.

[11] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *CVPR*. IEEE, 2009, pp. 248–255.

[12] Sergey Ioffe and Christian Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *ICML*, 2015, pp. 448–456.

[13] Nitish Srivastava, Geoffrey E Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov, "Dropout: a simple way to prevent neural networks from overfitting.," *Journal of Machine Learning Research*, vol. 15, no. 1, pp. 1929–1958, 2014.

[14] 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.

[15] Kevin Swersky Geoffrey Hinton and Nitish Srivastava, "rmsprop: Divide the gradient by a running average of its recent magnitude. coursera: Neural networks for machine learning," 2012.

[16] Gabriel J Brostow, Julien Fauqueur, and Roberto Cipolla, "Semantic object classes in video: A high-definition ground truth database," *Pattern Recognition Letters*, vol. 30, no. 2, pp. 88–97, 2009.

[17] S Hussain Raza, Matthias Grundmann, and Irfan Essa, "Geometric context from videos," in *CVPR*, 2013, pp. 3081–3088.

[18] Simon Jégou, Michal Drozdzal, David Vazquez, Adriana Romero, and Yoshua Bengio, "The one hundred layers tiramisu: Fully convolutional densenets for semantic segmentation," *arXiv preprint arXiv:1611.09326*, 2016.

[19] Abhijit Kundu, Vibhav Vineet, and Vladlen Koltun, "Feature space optimization for semantic video segmentation," in *CVPR*, 2016, pp. 3168–3175.

[20] Yuhang Wang, Jing Liu, Yong Li, Jun Fu, Min Xu, and Hanqing Lu, "Hierarchically supervised deconvolutional network for semantic video segmentation," *Pattern Recognition*, vol. 64, pp. 437–445, 2017.

[21] Du Tran, Lubomir Bourdev, Rob Fergus, Lorenzo Torresani, and Manohar Paluri, "Deep end2end voxel2voxel prediction," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, 2016, pp. 17–24.