Fully Convolutional Neural Networks with Full-scale-features for Semantic Segmentation [*]

Tianxiang Pan, Bin Wang, Guiguang Ding, Jun-Hai Yong
School of Software, Tsinghua University, Beijing 100084, China
ptx9363@gmail.com, {wangbins,dinggg,yongjh}@tsinghua.edu.cn

Abstract

In this work, we propose a novel method to involve full-scale-features into the fully convolutional neural networks (FCNs) for Semantic Segmentation. Current works on FCN has brought great advances in the task of semantic segmentation, but the receptive field, which represents region areas of input volume connected to any output neuron, limits the available information of output neuron's prediction accuracy. We investigate how to involve the full-scale or full-image features into FCNs to enrich the receptive field. Specially, the full-scale feature network (FFN) extends the full-connected network and makes an end-to-end unified training structure. It has two appealing properties. First, the introduction of full-scale-features is beneficial for prediction. We build a unified extracting network and explore several fusion functions for concatenating features. Amounts of experiments have been carried out to prove that full-scale-features makes fair accuracy raising. Second, FFN is applicable to many variants of FCN which could be regarded as a general strategy to improve the segmentation accuracy. Our proposed method is evaluated on PASCAL VOC 2012, and achieves a state-of-art result.

Introduction

Fully convolution network (FCN) has brought great breakthrough on semantic segmentation task (Long, Shelhamer, and Darrell 2015). By building the deep convolutional neural network (DCNN) fully convolutional, FCN could take input of arbitary size and produce pixel-wise predication. Recently, many FCN-based models are proposed and achieve striking results on semantic segmentation benchmarks (Chen et al. 2016; Lin et al. 2015; Zheng et al. 2015; Lin et al. 2016).

Looking into the current FCN models especially the failure examples, we found that most of the failure examples have large or full-image objects. In these examples, large objects cannot be efficiently detected because of the confined receptive field of FCN, as shown in Fig. 1. For enlarging the receptive field and enriching the prediction in-

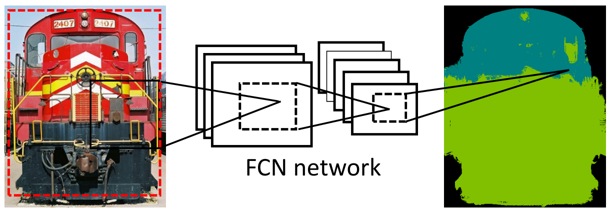

Figure 1: The limited receptivity of FCN cannot effectively segment the large objects. True scale of object is marked in the red bounding box.

formation, there are two main successful methods to obtain a better prediction results on large objects. The first one is employing multi-scale features from different layers of DCNN (Long, Shelhamer, and Darrell 2015) or from resized scale of input image to get a larger receptive field (Chen et al. 2016; Szegedy et al. 2014; Mostajabi, Yadollahpour, and Shakhnarovich 2015; Ding et al. 2016). The second type of methods is employing a global postprocessing technique, like Conditional Random Field (CRF) (Liu et al. 2015; Lin et al. 2015; Zheng et al. 2015), to fuse the entire image features.

By employing multi-scale features, FCNs could achieve a larger receptive field and make a more accurate prediction. Some of these FCNs (Long, Shelhamer, and Darrell 2015) concatenate the different layers of network, combining the deep, coarse layer's semantic information with shallow, fine layer's appearance information. This type of methods expand the receptive field in a certain extent (enlarged from 128 pixels to 224 pixels) but are still insufficient for large objects (width up to 400 or 500 pixels) filled in the whole image. Larger receptive fields will bring much more computation as well. By employing full-connected CRF (Chen et al. 2016), the final output results of FCN will be treated as a prediction prior and then be integrated for the whole image segmentation. Because this type of methods often take a two-step process, the training time is usually long.

From another aspect, detecting large-size objects is more like locating the objects instead of segmentation. The state-of-art object detection methods (Redmon et al. 2015; Ren et al. 2015; He et al. 2015; Jifeng Dai 2016) have achieved ex-

[*]This research was supported by the National Natural Science Foundation of China (Grant No. 61571269 and No. 61373071)

cellent performance by making full advantage of the Deep Convolutional Neural Network (DCNN) features. We notice that most of them utilize the information from the whole image, or namely full-scale feature, as an essential detection feature. Inspired by this observation, to overcome the hurdles for detecting large objects in semantic segmentation, we propose to integrate the full-scale-features network on FCNs in a unified network model.

In this paper, we model the full-scale-features in a full-connected way and use a sub-sampling network to get out of the dilemma of tremendous computation from full-connected layer. A global-to-local scale transform network is designed to concatenate the different features. An advantage of our method is that our full-scale-features extracting network (FFN) is independent, which means the FFN could be applicable and effective to most of the FCN-based network.

As best as we know, there's only one previous work trying to utilize the full-scale image features (Liu, Rabinovich, and Berg 2015) to improve segmentation accuracy. They use global pooling method to generate global or full-scale features, replicate the global feature into target size, and then apply L2 Norm to fuse these different features. For further exploiting the global contextual information, we take a different network to model the full-scale features which can produce a grid-based prediction feature maps. In this way, we could not only synthesize the whole image but also give a regional-based full-scale features. We design a multi-layer scale transform module to concatenate the different types of features. Finally, our method outperforms Liu's model and make a state-of-art prediction result on PASCAL VOC 2012 dataset.

The main contributions of this paper are as follows:

- We propose an end-to-end network FFN, to extract the full-scale-features in FCNs for semantic segmentation.
- We propose a module for transforming the various scale's features into an unified network.
- The proposed full-scale-features network (FFN) can be easily adapted for enhancing other FCN-based method and improving the prediction accuracy.
- Our method outperforms the other full-scale or global-feature methods and achieves a state-of-art result on the Pascal VOC 2012.

In the next section, we review related work on FCNs with multi-scale features and FCNs with CRFs. Some object detection methods are also reviewed for employing the full-scale features in the related work. Model Architecture section introduces the proposed method and the performance evaluation is shown in Experiments section. Finally, we make a conclusion and present more results in final section.

Related work

Our method draws on the success of FCN with its improved models for multi-scale features and CRF finetuning. Object detection models also inspire the design of full-scale-features network.

Fully Convolutional Networks

By replacing the fully-connected layer of Classcification Neural Network (CNN) with convolution layer, Fully convolutional Network (FCN) could make a pixel-level classification prediction (Long, Shelhamer, and Darrell 2015). For the semantic segmentation task, FCN and its variants have demonstrated state-of-art performance. In particular, our networks are experimented on the original FCN and Deeplab models (Chen et al. 2016; Liu et al. 2015). Amounts of further improvement on FCNs are mainly focused on the multi-scale feature learning or using CRF to refine the segmentation results.

Multi-Scale Features

It is well-known that multi-scale features are beneficial for object detection and segmentation in computer vision. The original FCN-8s defines a skip architecture that combines the deep layer with shallow layer to refine the coarse prediction into detailed segmentation. DeepLab-MSc (Chen et al. 2016) proposed a MultiLayer Perceptrons (MLP) to extract multi-scale features, and its improved model, DeepLab-Attention (Chen et al. 2015), trained an attention-to-scale model to learn a softly weighted multi-scale features. (Liu et al. 2015) resized the input image with different scales and fuse these multi-scale features for Conditional Rendom Field's (CRF) unary prior. (Mostajabi, Yadollahpour, and Shakhnarovich 2015) employed the "zoom-out" features that maps the superpixels to rich feature representations extracted from its nested regions.

Full-scale Features

Different from the multi-scale features, full-scale features are required to integrate the whole feature maps. ParseNet (Liu, Rabinovich, and Berg 2015) employs global-pooling, L2 normalization and simple replicated unpooling for aggregating the whole-image's features to provide global contextual information. Our method takes a similar pipeline with Liu's method but the differences between ParseNet with our full-scale-features network are significant. Instead of simply replicating the global features into pixels, we employ a gird-based global-features training structure and, further more, design a multi-layer fusion network to concatenate the full-scale features. Our method also performs better prediction result than ParseNet. More explanations and details will be discussed in Model Architecture Section. (Dai, He, and Sun 2015) proposed a multi-task learning method for instance-aware semantic segmentation. In its model, the region-scale features are extracted in a full-connected way for segmentation mask learning. Our method builds a similar feature extracting network but for the whole-image features.

Conditional Random Field

Another method to fuse the whole image features is applying a Conditional Random Field (CRF) to the FCN's prediction results which could involve both context and pixel information. (Chen et al. 2014) first proposed to use CRF for finetuning the semantic segmentation results. They treated the

CRF as an independent postprocess and the prediction results of FCN are incorporated as unary prior of CRF. Since this method takes a two-step training process, many following works are focused on the unified training model for FCN and CRF. (Liu et al. 2015) proposed a CNN-Based model solving the CRF optimization iteration through one pass of network forwarding. (Zheng et al. 2015) modeled the optimization iterations of CRF as an RNN (Recurrent Neural Networks) network and thus they made an end-to-end training method for FCN with CRF. They also utilized the message passing to learn the CRF inference in their unified model. By involving more superpixels and object-detection prior to CRF, (Lin et al. 2015) proposed a piece-wise training method for pixel-level prediction.

Object detection

Current advance on object detection is another inspiration for our model. Faster RCNN (Ren et al. 2015) employed a SPP-Net to detect different scale and ratio objects in an image. SPP-Net (He et al. 2014) employed different shape of convolution kernels and thus could help detect multi-scale and multi-ratio objects. Another successful object detection model is YOLO (Redmon et al. 2015), which treats the localization task as a regression problem. It takes an FCN-like network to extract basic features, and divides the whole image input several grid. YOLO predicts whether the center of an object falls into a grid cell and uses the basic features to predict the specific bounding box's size and location. Inspired by YOLO, we split the middle feature map into grid too. We reframe the full-scale feature learning task into a low-resolution segmentation task but the output is feature vectors instead of feature maps. The low-resolution but full-scale features are further convoluted and fused with FCN prediction to generate a context-aware result.

Model Architecture

This section discusses the model architecture of our Full-Scale-Features Network (FFN), including the feature extracting model and the fusion function design.

Full Scale Features Extracting

The specific network architecture for Full-Scale Features Network (FFN) is shown in Fig. 2. The key idea of generating the full-scale features is including a fully-connected layer to fuse the whole image feature map, but the complexity and computation costs make the combination hard to work. In our method, a fix-sized sub-pooling is employed to project the big CNN features space into a small, acceptable feature space. After the pooling layer, two full-connected layers are designed to calculating the final full-scale features. To concatenate the full-scale features, a backward-remap layer is required to reshape the feature vectors into feature maps.

The fully-convolutional feature maps, which are generated after amounts of convolution layers, can be of arbitrary size based on its subsampling coefficients and the size of input image while the fully-connected layers require fixed-size input features. This transformation from arbitrary feature map to fixed maps has been explored by spatial pyramid pooling. Our fixed-size subpooling takes a similar form of network but for semantic segmentation instead of object detection. In this way, we divide the final full-scale feature maps into $S \times S$ grid, and the result maps will have C channels (S = 7 and C = 21 in our experiments). These setups mean that we will obtain the full-scale feature maps with size of $21 \times 7 \times 7$ and the full-scale feature vectors with size of 1029 ($= 21 \times 7 \times 7$).

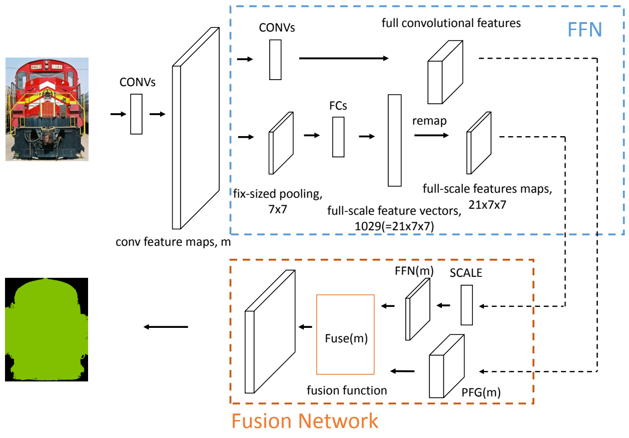

Figure 2: Full Scale Features extracting network.

Looking into the full-scale features extracting submodule, we found that this sub-network is actually independent of the semantic segmentation which means the FFN could draw full-scale features from any convolutional feature maps. For another independent aspect, the FFN could be supervised by a low-resolution segmentation label that makes the network multi-task. This selection will be experimented comparing with the no-supervised FFN in the following section.

Feature Fusion

After getting the these full-scale features, We explore the fusing functions for integrating with original FCN's local-scale features. Because of the fix-sized sub-sampling layer, the FFN's output features is downsampled and coarse. In order to fuse these coarse full-scale features with high-resolution FCN's features effectively, we are required to make a concatenation and refinement layer.

Notation We denote by $\boldsymbol{x}$ the image values and $\boldsymbol{y}$ the segmentation final predication after deconvolution and unpooling layers. The middle layer features which are generated before deconvolution and after convolution are denoted as $\boldsymbol{m}$, more specifically for example the output of conv5 of VGG16 network. As mentioned before, our **f**ull scale **f**eatures extracting **n**etwork could be indenpendent, so that the network output can be denoted as an independent function of $\boldsymbol{m}$: $\boldsymbol{FFN(m)}$. Some of the FCN variants also have a post-feature generating network, such as the fc6, fc7 layer in VGG16 network, and we denote the prediction result after these **p**ost-**f**eature **g**enerating networks as $\boldsymbol{PFG(m)}$.

Before concatenating the different types of features, we are first required to normalize the features. Unnormalized features training can lead to weight unbalance between the large scale features and small scale features. Inspired by ParseNet, we include a scale layer after each feature to learn the adequate transforming scales for each type of feature. A bilinear interpolation layer is added to unpool the fix-size feature maps ($21 \times 7 \times 7$ in our experiments) onto the FCN feature maps size.

After normalization, our model is trying to fuse the full-scale features with FCN results. A trivial design of fusion function (denoted as ***Fuse(m)***) could be formulate of,

$$Fuse(m) = FFN(m) + PFG(m) \tag{1}$$

where the '+' represents the element-wise sum of each feature.

Note that the *FFN(m)* could be regarded as a middle latent features for the final pixel results. Concatenating the middle features with each other could reasonably make a better prediction than the trivial design, and the formulation become,

$$Fuse(m) = FFN(m) \,.\!+\, PFG(m) \tag{2}$$

where the '.+' represents the channel concatenation of each feature.

While the post-feature generating network is not necessary for some FCN variants, these models will directly use the extracting features to make pixel classification. The *PFG(m)* term is not available in the situation. For integrating the full-scale features into these models, we modified the fusion function into an iteration way,

$$Fuse(m) = FFN(m) \,.\!+\, m \tag{3}$$

Experiments have shown that this type of fusion function make similar amount of accuracy improvement for semantic segmentation with function (2). Inspired by this function transform, we finally proposed a multi-layer fusion function to explore the detection ability of full-scale features. The formulation could become like,

$$Fuse(m) = FFN(m) \,.\!+\, m \,.\!+\, PFG(m) \tag{4}$$

These fusion functions are illustrated in Fig. 3.

In experiments, we contrast these fusion functions on PASCAL VOC val dataset and employ the network in Fig. 3(d) for its best performance in our final architecture to concatenate the full-scale features. More specifically, we fuse the full-scale feature with the fcn feature firstly and then concatenate with the output of deeper fully-convolutional networks. The final concatenated features are utilized for predicting segmentation results.

Training and Loss Function

Our full-scale extracting network are basically built on the FCN models. So that the models of our method are all pretrained by the original FCN model or it variants. For adding the FFN into FCNs gracefully, we adopt a two-step training method which firstly learns the FFN parameters with FCN's parameters fixed. After amounts of iterations, the second

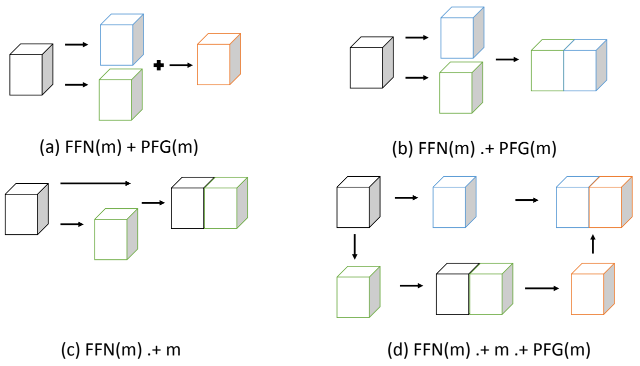

Figure 3: Fusion Functions for concatenating the full-scale features.

step train the whole unified network including both the convolutional and connected layers to improve prediction accuracy.

FFN predict the final pixel-level classification results for semantic segmentation. But as we referred in the model architecture section, the independent result of full-scale features results could be treated as a sub-pooled segmentation task. In our setups, the segmentation task only need to predict the 7×7 grid of image labels. By specific denotation, these singe-task training and multi-task network's loss function could formulated as,

$$Loss(m) = L1(Fuse(m)) \tag{5}$$

and

$$Loss(m) = L1(Fuse(m)) + L2(FFN(m)) \tag{6}$$

Here, $L1$, $L2$ represent the loss term of concatenating features results and full-scale feature results. The comparison experiments for the different loss functions will be carried out in the next section.

Comparison to ParseNet

Our method builds a similar pipeline with ParseNet for integrating the full-scale feature but the specific modeling are quite different. The ParseNet used the replicating full-scale features and we employ a full-connected grid-based full-scale features. We investigate the possible fusion functions for concatenating different scale of features instead of the concat-feature fusion described in ParseNet. In ParseNet, the fusion functions is fixed to concatenation but the Fusion periods are classified as early fusion, concatenate the full-scale features directly, and late fusion which adds an additional layer to full-scale features first. In our method, we tried to enlarge the detail information of full-scale features by refusing the **FFN(m)** with **m**. This refusing function is shown to be more effective than simple concatenation. And for the results, our method outperforms ParseNet equally evaluated based on the DeepLab-LargeFOV model for PASCAL VOC 2012.

Experiments

In this section, we mainly evaluate our method on VOC 2012 benchmark dataset (Everingham et al. 2010). The VOC 2012 dataset has 20 object classes with an extra background class. The original datasets are labeled with 1464 train images and 1449 validation images. (Hariharan et al. 2011) labels additional 9118 images with segmentation annotations which are commonly used for training semantic segmentation models. We use these additional annotations and test our method on both validation and test datasets.

We use the mean intersection over union (Mean IoU) results as the standard evaluation score to compare our method with others. For fairly comparing our improved models with original FCN-based models, we have reproduced all of the FCN-based segmentation model in our experiments and trained these methods from their pre-trained models such as VGG16 classification models (Chatfield et al. 2014).

SGD with mini-batch is used for training. A training mini-batch of 10 images is taken and the initial learning rate is 1e-9 with a step learning policy that multiplied by 0.1 each 20k iterations. We use the momentum of 0.9 and the weight decay of 0.0005. Fine-tuning the model on a NVIDIA GeForce TiTan GPU costs about 1 to 1.5 days. Our full-scale features model is implemented in the Caffe framework and will be published soon.

FCN-based Baselines

Since our FFN network is a promoting network, we are required to reproduce the FCN-based basic models firstly so as to append our FFN to these model. We make experiments on the original FCN models and Deeplab models.

FCN-8s is the first and the basic model we utilize for experiments. The FCN-8s model transfers the full-connected layers into fully-convolution and employs a skip net to combine the deep layer's prediction with shallow layer's features. By being aware of the shallow layer, FCN-8s could predict in a finer resolution which performs 1/8 size of original image. In the reproduction of FCN-8s, we found that bilinear-interpolation uppooling layers not only speeds up the training but also makes a better prediction result. Yet after all-at-once training of FCN-8s, this model achieves 62.7% mean IoU accuracy of validation dataset, that is 2.6% lower than the reported accuracy (Long, Shelhamer, and Darrell 2015). The reason of the accuracy gap between our reproduction and published results probably is the lack of more training iteration (we train the network for 60k iterations) and a smaller learning rate. Because our full-scale features model only makes a promotion to the current FCN models, it's still fair for us to evaluate our method on the reproduction FCN networks.

DeepLab is another network we experiment. DeepLab employs the same fully-convolutional network structure for segmentation and uses a full-connected CRF as postprocessing to finetune the results. The DeepLab-LargeFOV enlarges the receptive field of Deeplab and employs a smaller kernel size of 3×3 for the last prediction layer in VGG16 network, which makes it easier to converge and train. We reproduced the DeepLab-LargeFOV model and achieved a 67% mean IoU accuracy of validation dataset which is comparable with the reported results of (Chen et al. 2016).

Results and Analysis

We have experimented our method on PASCAL VOC 2012 validation dataset. For implementing details, We set the full-scale features map size to be $21 \times 7 \times 7$, and use two full-connected layers with 1024 neurons. We evaluate the different fusion functions firstly on FCN-8s. We train the FCN-8s in an all-at-once method which is pre-trained on the published model from (Long, Shelhamer, and Darrell 2015). Because our trainings are carried on the additional annotation dataset, the validation dataset is slitted into trained images and non-trained images. Only the non-trained imageset which contains 736 images are validated on our model.

PASCAL VOC Validation	Mean IoU
FCN-8s	62.7%
FCN-8s-FFN(a)	57.4%
FCN-8s-FFN(b)	64.2%
FCN-8s-FFN(d)	**64.5%**

Table 1: Results for different fusion functions for FCN-8s with FFN on the PASCAL VOC 2012 validation dataset. FCN-8s-FFN(a), (b), (d) are the models with different fusion functions shown in Fig. 3.

From Table 1, we can see that the introduction of our full-scale feature network is basically beneficial. In further, the concatenating fusion functions are outperforming the simply summation of different features. In particular, we observe that the summation fusing function, FCN-8s-FFN (a), gets a worse accuracy than FCN-8s baseline. The fail of performance probably comes from the low-resolution FFN middle prediction thus causing a big interference in the summation fusion functions.

Because of the successful reproduction of DeepLab and DeepLab-LargeFOV, we conduct more experiments on the DeepLab models. When training the DeepLab models and the FFN extensions, a mini-batch of 5 images and 60k iterations of training are conducted. Because of the poor result on the FCN-8s model, we don't prefer to experiment the fusion function (a) on DeepLab anymore. Tabel 2 shows the Deeplab results on the same validation dataset. For a fair comparison, all of the DeepLab-based models are post-processed by a independent full-connected CRF layer (Koltun 2011) to finetune the predictions. The CRF parameters are cross-validated in validation dataset to maximize the accuracy.

The sub-Supervision experiments on the FFN are conducted on the validation dataset. For supervised learning for full-scale features with size of $21 \times 7 \times 7$, we see it as a 7×7 grid classification problem and the 21 channels just correspond to 21 different object classes in PASCAL VOC 2012 dataset. While adding the additional Supervision results in a slight decrease in the prediction accuracy as shown in Table. 2 . we think the extracted full-scale features should not be regarded as prediction result but a latent variables which need to be further fused into local-features.

PASCAL VOC Validation	Mean IoU
DeepLab-LargeFOV	67.7%
DeepLab-LargeFOV-FFNwithSuper	67.4%
DeepLab-LargeFOV-FFN(b)	69.3%
DeepLab-LargeFOV-FFN(d)	**69.7%**

Table 2: Results for DeepLab with FFN on the PASCAL VOC 2012 validation dataset.

As shown in Table 2, the Deep-Lab-FFN(b) and (d) achieve 2% accuracy promotion for the base DeepLab model. For fairly comparing our method with former method, we train the DeepLab-FFN network for VOC 2012 test dataset, and make a broader comparison. Note that our Deep-Lab-FFN is only trained on the PASCAL VOC 2012 train and val datasets with additional annotations from (Hariharan et al. 2011), so that we don't compare our method with MSCOCO dataset (Lin et al. 2014) finetuned models or Resnet-based models. These two methods have been proved to be a great help for semantic segmentation task. Our FFN can be performed as a submodule for generating full-scale features for these methods and datasets. More experiments on these state-of-art methods will be conducted in our future work.

PASCAL VOC Test 2012	Mean IoU
Zoom-out	69.6%
DPN* (Liu et al. 2015)	74.1%
RNN* (Zheng et al. 2015)	72.0%
Piecewise* (Lin et al. 2015)	70.7%
DeepLab-LargeFOV (Baseline)	70.3%
DeepLab-LargeFOV-MSc	71.6%
DeepLab-LargeFOV-ParseNet	69.8%
DeepLab-LargeFOV-FFN(b)	71.2%
DeepLab-LargeFOV-FFN(d)	71.3%

Table 3: Results Comparison experiments on VOC 2012 test dataset. The segmentation accuracy results are provided from PASCAL VOC evaluation server leaderboard or published by the author. We only compare the models training on the trainval dataset with additional annotation. Approaches unify the CRF-learning are marked with *.

We compare our final FFN models with other state-of-art methods in Table 3. Firstly we focus on the comparing with ParseNet, for their similar pipelines and motivation. The results reveal that our method outperforms the ParseNet which makes it a more effective approach to involving the full-scale image features. Secondly from considering our method with the comparable DeepLab-LargeFOV-MSc which resizes the input images and produces a multi-scale feature, we notice that involving the full-scale image feature makes a similar amount of improvement with multi-scale features. Yet the multi-scale features and full-scale features are not conflicting. Our full-scale features could be added to each resized feature of the multi-scale features to enrich the final prediction.

By comparing with the other state-of-art methods, we found that the unified CRF training methods are outperforming the single FCNs or two-step training with CRF. The independence of our FFN makes it applicable to all of these method and only need to add only a little and controllable extra computation for the network.

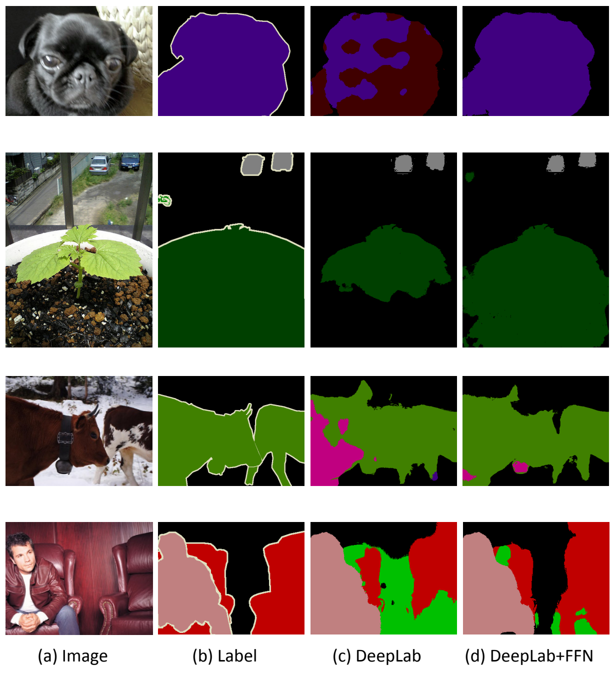

Figure 4: More results comparing the DeepLab agianst DeepLab-FFN.

Conclusion

We proposed Full-Scale Feaures Network (FFN) to involving the whole-image features into semantic image segmentation, which has two appealing properties. First, FFN is proven effective for learning the full-scale features which could be utilized for improving the FCN-based segmentation methods while recognizing large objects. Second, the independence of network module enables our method to finetune most of the current FCN-based methods. Adding the sub-network FFN to learning full-scale features only costs a little and controllable amount of computation. More results are shown in Fig. 4.

The DeepLab with FFN achieves state-of-art performance on VOC2012 trained only by trainval dataset with additional annotation and our method outperforms the previous works on involving full-scale features. Future directions include experimenting FFN onto more multi-scale or CRF unifying methods, making the full-scale features a single and independently trainable FFN subnetwork to effectively make up a multi-task model.

References

Chatfield, K.; Simonyan, K.; Vedaldi, A.; and Zisserman, A. 2014. Return of the devil in the details: Delving deep into convolutional nets. *arXiv preprint arXiv:1405.3531*.

Chen, L.-C.; Papandreou, G.; Kokkinos, I.; Murphy, K.; and Yuille, A. L. 2014. Semantic image segmentation with deep convolutional nets and fully connected crfs. *arXiv preprint arXiv:1412.7062*.

Chen, L.-C.; Yang, Y.; Wang, J.; Xu, W.; and Yuille, A. L. 2015. Attention to scale: Scale-aware semantic image segmentation. *arXiv preprint arXiv:1511.03339*.

Chen, L.-C.; Papandreou, G.; Kokkinos, I.; Murphy, K.; and Yuille, A. L. 2016. Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. *arXiv preprint arXiv:1606.00915*.

Dai, J.; He, K.; and Sun, J. 2015. Instance-aware semantic segmentation via multi-task network cascades. *arXiv preprint arXiv:1512.04412*.

Ding, G.; Guo, Y.; Zhou, J.; and Gao, Y. 2016. Large-scale cross-modality search via collective matrix factorization hashing. *IEEE Transactions on Image Processing* 25(11):5427–5440.

Everingham, M.; Van Gool, L.; Williams, C. K.; Winn, J.; and Zisserman, A. 2010. The pascal visual object classes (voc) challenge. *International journal of computer vision* 88(2):303–338.

Hariharan, B.; Arbeláez, P.; Bourdev, L.; Maji, S.; and Malik, J. 2011. Semantic contours from inverse detectors. In *2011 International Conference on Computer Vision*, 991–998. IEEE.

He, K.; Zhang, X.; Ren, S.; and Sun, J. 2014. Spatial pyramid pooling in deep convolutional networks for visual recognition. In *European Conference on Computer Vision*, 346–361. Springer.

He, K.; Zhang, X.; Ren, S.; and Sun, J. 2015. Deep residual learning for image recognition. *arXiv preprint arXiv:1512.03385*.

Jifeng Dai, Yi Li, K. H. J. S. 2016. R-FCN: Object detection via region-based fully convolutional networks. *arXiv preprint arXiv:1605.06409*.

Koltun, V. 2011. Efficient inference in fully connected crfs with gaussian edge potentials. *Adv. Neural Inf. Process. Syst.*

Lin, T.-Y.; Maire, M.; Belongie, S.; Hays, J.; Perona, P.; Ramanan, D.; Dollár, P.; and Zitnick, C. L. 2014. Microsoft coco: Common objects in context. In *European Conference on Computer Vision*, 740–755. Springer.

Lin, G.; Shen, C.; Reid, I.; et al. 2015. Efficient piecewise training of deep structured models for semantic segmentation. *arXiv preprint arXiv:1504.01013*.

Lin, Z.; Ding, G.; Han, J.; and Wang, J. 2016. Cross-view retrieval via probability-based semantics-preserving hashing. *IEEE Transactions on Cybernetics*.

Liu, Z.; Li, X.; Luo, P.; Loy, C.-C.; and Tang, X. 2015. Semantic image segmentation via deep parsing network. In *Proceedings of the IEEE International Conference on Computer Vision*, 1377–1385.

Liu, W.; Rabinovich, A.; and Berg, A. C. 2015. Parsenet: Looking wider to see better. *arXiv preprint arXiv:1506.04579*.

Long, J.; Shelhamer, E.; and Darrell, T. 2015. Fully convolutional networks for semantic segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 3431–3440.

Mostajabi, M.; Yadollahpour, P.; and Shakhnarovich, G. 2015. Feedforward semantic segmentation with zoom-out features. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 3376–3385.

Redmon, J.; Divvala, S.; Girshick, R.; and Farhadi, A. 2015. You only look once: Unified, real-time object detection. *arXiv preprint arXiv:1506.02640*.

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

Szegedy, C.; Reed, S.; Erhan, D.; and Anguelov, D. 2014. Scalable, high-quality object detection. *arXiv preprint arXiv:1412.1441*.

Zheng, S.; Jayasumana, S.; Romera-Paredes, B.; Vineet, V.; Su, Z.; Du, D.; Huang, C.; and Torr, P. H. 2015. Conditional random fields as recurrent neural networks. In *Proceedings of the IEEE International Conference on Computer Vision*, 1529–1537.