PARSENET: LOOKING WIDER TO SEE BETTER

Wei Liu
UNC Chapel Hill
wliu@cs.unc.edu

Andrew Rabinovich
MagicLeap Inc.
arabinovich@magicleap.com

Alexander C. Berg
UNC Chapel Hill
aberg@cs.unc.edu

ABSTRACT

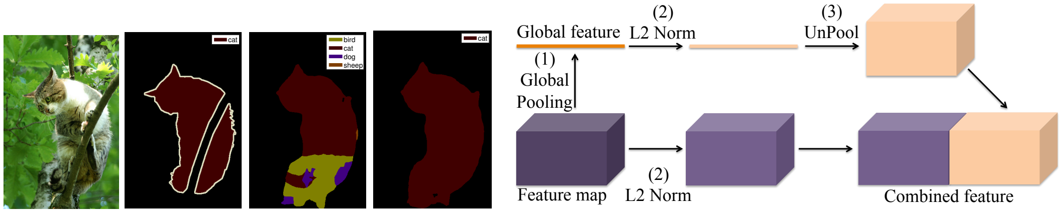
We present a technique for adding global context to fully convolutional networks for semantic segmentation. The approach is simple, using the average feature for a layer to augment the features at each location. In addition, we study several idiosyncrasies of training, significantly increasing the performance of baseline networks (e.g. from FCN Long et al. (2014)). When we add our proposed global feature, and a technique for learning normalization parameters, accuracy increases consistently even over our improved versions of the baselines. Our proposed approach, ParseNet, achieves state-of-the-art performance on SiftFlow and PASCAL-Context with small additional computational cost over baselines, and near state-of-the-art performance on PASCAL VOC 2012 semantic segmentation with a simple approach. Code is available at https://github.com/weiliu89/caffe/tree/fcn .

1 INTRODUCTION

Semantic segmentation, largely studied in the last 10 years, merges image segmentation with object recognition to produce per-pixel labeling of image content. The currently most successful techniques for semantic segmentation are based on fully convolution networks (FCN) Long et al. (2014). These are adapted from networks designed to classify whole images Krizhevsky et al. (2012); Szegedy et al. (2014a); Simonyan & Zisserman (2014), and have demonstrated impressive level of performance. The FCN approach can be thought of as sliding an classification network around an input image (by sharing computation), and processes each sliding window area independently. In particular, FCN disregards global information about an image, thus ignoring potentially useful scene-level semantic context. In order to integrate more context, several approaches Chen et al. (2014); Schwing & Urtasun (2015); Lin et al. (2015); Zheng et al. (2015), propose using techniques from graphical models such as conditional random field (CRF), to introduce global context and structured information into a FCN. Although powerful, these architectures can be complex, combining both the challenges of tuning a deep neural network and a CRF, and require a fair amount of experience in managing the idiosyncrasies of training methodology and parameters. At the least, this leads to time-consuming training and inference.

In this work, we propose *ParseNet*, an end-to-end simple and effective convolutional neural network, for semantic segmentation. One of our main contributions, as shown in Fig. 1, is to use global context to help clarify local confusions. Looking back at previous work, adding global context for semantic segmentation is not a new idea, but has so far been pursued in patch-based frameworks Lucchi et al. (2011). Such patch-based approaches have much in common with detection and segmentation work that have also shown benefits from integrating global context into classifying regions or objects in an image Szegedy et al. (2014b); Mostajabi et al. (2014); Mottaghi et al. (2014). Our approach allows integrating global context in an end-to-end fully convolutional network (as opposed to a patch-based approach) for semantic segmentation with small computational overhead. In our setting, the image is not divided into regions or objects, instead the network makes a joint prediction of all pixel values. Previous work on fully convolutional networks did not include global features, and there were limits in the pixel distance across which consistency in labeling was maintained.

The key ”widget” that allows adding global context to the FCN framework is simple, but has several important consequences in addition to improving the accuracy of FCN. First, the entire end-to-end process is a single deep network, making training relatively straightforward compared to combining deep networks and CRFs. In addition, the way we add global context does not introduce much com-



(a) Image (b) Truth (c) FCN (d) ParseNet (e) ParseNet contexture module overview.

Figure 1: ParseNet uses extra global context to clarify local confusion and smooth segmentation.

putational overhead versus training and evaluating a standard FCN, while improving performance significantly. In our approach, the feature map for a layer is pooled over the whole image to result in a context vector. This is appended to each of the features sent on to the subsequent layer of the network. In implementation, this is accomplished by unpooling the context vector and appending the resulting feature map with the standard feature map. The process is shown in Fig. 1. This technique can be applied selectively to feature maps within a network, and can be used to combine information from multiple feature maps, as desired. Notice that the scale of features from different layers may be quite different, making it difficult to directly combine them for prediction. We find that L_2 normalizing features for each layer and combining them using a scaling factor learned through backpropagation works well to address this potential difficulty.

In section 3, we demonstrate that these operations, appending global context pooled from a feature map along with an appropriate scaling, are sufficient to significantly improve performance over the basic FCN, resulting in accuracy on par with the method of Chen et al. (2014) that uses detailed structure information for post processing. That said, we do not advocate ignoring the structure information. Instead, we posit that adding the global feature is a simple and robust method to improve FCN performance by considering contextual information. In fact, our network can be combined with explicit structure output prediction, e.g. a CRF, to potentially further increase performance.

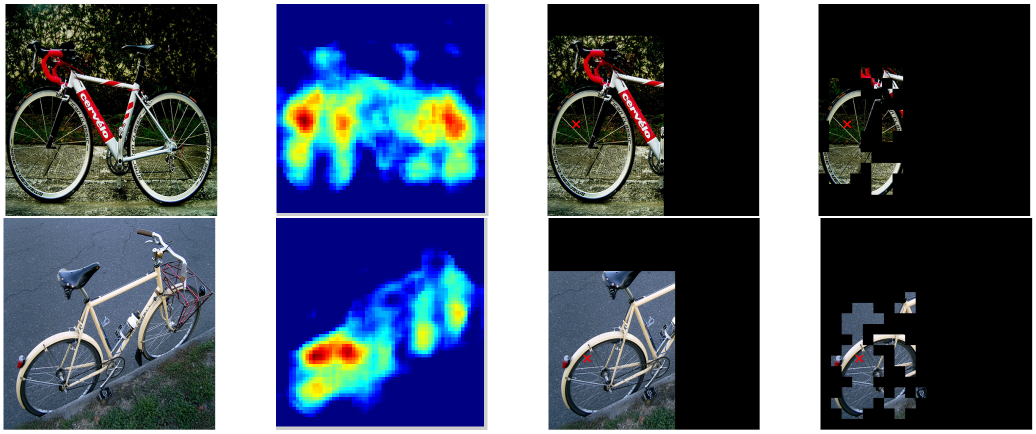
2 ParseNet

2.1 Global Context

Context is known to be very useful for improving performance on detection and segmentation tasks using deep learning. Mostajabi et al. (2014); Szegedy et al. (2014b) and references therein illustrate how context can be used to help in different tasks. As for semantic segmentation, per pixel classification, is often ambiguous in the presence of only local information. However, the task becomes much simpler if contextual information, from the whole image, is available.

It is known that features from the top layers of a network have very large receptive fields (e.g. fc7 in FCN with VGG has a 404×404 pixels receptive field), it is theoretically possible for an FCN to learn to capture context information if that is useful. However, in practice, we observe that the network learns to be ”myopic” (e.g. toward object parts) and to ignore bits of background. The empirical size of the receptive fields is much smaller, and is not enough to capture the global context. Following Zhou et al. (2014), we slide a small patch of random noise across the input image, and measure the change in the activation of the desired layer. If the activation does not vary significantly, that suggests the given random patch is outside of the empirical receptive field, as shown in Figure 2. The effective receptive field at the last layer of this network barely covers $\frac{1}{4}$ of the entire image. Fortunately, it is rather straightforward to get the context within the FCN architecture. Specifically, we use global average pooling and pool the context features from the last layer or any layer if that is desired. The quality of semantic segmentation is greatly improved by adding the additional global feature to local feature map, either with early fusion [1] or late fusion as discussed in Sec. 2.2. For example, Fig 1 has misclassified a large portion of the image as bird since it only used local information, however, adding contextual information in the loop, which might contain

[1] we use unpool operation by simply replicating the global feature horizontally and vertically to have the same size as the local feature map.



(a) Original Image (b) Activation map (c) Theoretical RF (d) Empirical RF

Figure 2: **Receptive field (RF) size for last layer.** (a) original image; (b) activation map on bicycle from a channel of the last layer of a network; (c) theoretical receptive field of the maximum activation (marked by red cross) is defined by the network structure; (d) empirical receptive field affecting the activation. Clearly empirical receptive field is not large enough to capture the global context.

strong signal of cat, corrects the mistake. Experiment results on VOC2012 and PASCAL-Context dataset also verify our assumption. Compared with Chen et al. (2014), the improvement is similar as of using CRF to post-process the output of FCN.

In addition, we also tried to follow the spatial pyramid idea Lazebnik et al. (2006) to pool features from increasingly finer sub-regions and attach them to local features in the sub-regions, however, we did not observe significant improvements. We conjecture that it is because the (empirical) receptive field of high-level feature maps is larger than or similar as those sub-regions. However features pooled from the whole image are still beneficial.

2.2 Early Fusion and Late Fusion

Once we get the global context feature, there are two general standard paradigms of using it with the local feature map. First, the *early fusion*, illustrated in in Fig. 1 where we unpool (replicate) global feature to the same size as of local feature map spatially and then concatenate them, and use the combined feature to learn the classifier. The alternative approach, is *late fusion*, where each feature is used to learn its own classifier, followed by merging the two predictions into a single classification score Long et al. (2014); Chen et al. (2014). There are cons and pros for both fusion methods. If there is no additional processing on combined features, *early fusion* is quite similar to *late fusion* as pointed out in Hariharan et al. (2014). With *late fusion*, there might be a case where individual features cannot recognize something but combining them may and there is no way to recover from independent predictions. Our experiments show that both method works more or less the same if we normalize the feature properly for early fusion case.

When merging the features, one must be careful to normalize each individual feature to make the combined feature work well; in classical computer vision this is referred as the cue combination problem. As shown in Fig. 3, we extract a feature vector at a position combined from increasing higher level layers (from left to right), with lower level feature having a significantly larger scale than higher level layers. As we show in Sec. 3.2, by naively combining features, the resultant feature will not be discriminative, and heavy parameter tuning will be required to achieve sufficient accuracy. Instead, we can first L_2 normalize each feature and also possibly learn the scale parameter, which makes the learning more stable. We will describe more details in Sec. 2.3.

2.3 L_2 Normalization Layer

As discussed above and shown in Fig. 3, we need to combine two (or more) feature vectors, which generally have different scale and norm. Naively concatenating features leads to poor performance

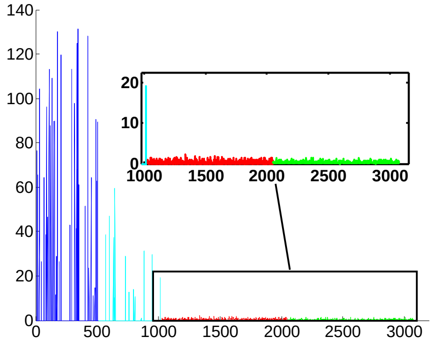


Figure 3: Features from 4 different layers have activations that are of drastically different scales. Each color corresponds to a different layers' feature. While *blue* and *cyan* are on a comparable scale, *red* and *green* features are of a scale 2 orders of magnitude less.

as the ”larger” features dominate the ”smaller” ones. Although during training, the weight might adjust accordingly, it requires very careful tuning of parameters and depends on dataset, thus goes against the robust principle. We find that by normalizing each individual feature first, and also learn to scale each differently, it makes the training more stable and improves performance.

L_2 norm layer is not only useful for feature combination. As was pointed out above, in some cases *late fusion* also works equally well, but only with the help of L_2 normalization. For example, if we want to use lower level feature to learn classifier, as demonstrated in Fig. 3, some of the features will have very large norm. It is not trivial to learn with it without careful weight initialization and parameter tuning. A work around strategy is to apply an additional convolutional layer Chen et al. (2014); Hariharan et al. (2014) and use several stages of fine-tuning Long et al. (2014) with much lower learning rate for lower layer. This again goes against the principle of simply and robustness. In our work, we apply L_2-norm and learn the scale parameter for each channel before using the feature for classification, which leads to more stable training.

Formally, let ℓ be the loss we want to minimize. Here we use the summed softmax loss. For a layer with d-dimensional input $\mathbf{x} = (x_1 \cdots x_d)$, we will normalize it using L_2-norm[2] with $\hat{\mathbf{x}} = \frac{\mathbf{x}}{||\mathbf{x}||_2}$ where $||\mathbf{x}||_2 = \left(\sum_{i=1}^{d} |x_i|^2 \right)^{1/2}$ is the L_2 norm of $\mathbf{x}$.

Note that simply normalizing each input of a layer changes the scale of the layer and will slow down the learning if we do not scale it accordingly. For example, we tried to normalize a feature s.t. L_2-norm is 1, yet we can hardly train the network because the features become very small. However, if we normalize it to e.g. 10 or 20, the network begins to learn well. Motivated by batch normalization Ioffe & Szegedy (2015) and PReLU He et al. (2015), we introduce a scaling parameter γ_i, for each channel, which scales the normalized value by $y_i = \gamma_i \hat{x}_i$.

The number of extra parameters is equal to total number of channels, and are negligible and can be learned with backprogation. Indeed, by setting $\gamma_i = ||\mathbf{x}||_2$, we could recover the L_2 normalized feature, if that was optimal. Notice that this is simple to implement as the normalization and scale parameter learning only depend on each input feature vector and do not need to aggregate information from other samples as batch normalization does. During training, we use backpropagation and chain rule to compute derivatives with respect to scaling factor γ and input data $\mathbf{x}$

$$\frac{\partial \ell}{\partial \hat{\mathbf{x}}} = \frac{\partial \ell}{\partial \mathbf{y}} \cdot \gamma \qquad \frac{\partial \ell}{\partial \mathbf{x}} = \frac{\partial \ell}{\partial \hat{\mathbf{x}}} \left(\frac{\mathbf{I}}{||\mathbf{x}||_2} - \frac{\mathbf{x}\mathbf{x}^T}{||\mathbf{x}||_2^3} \right) \qquad \frac{\partial \ell}{\partial \gamma_i} = \sum_{y_i} \frac{\partial \ell}{\partial y_i} \hat{x}_i \tag{1}$$

For our case, we need to do L_2-norm per each pixel in a feature map instead of the whole. We can easily extend the equations by doing it elemental wise as it is efficient.

[2] We have only tried L_2 norm, but can also potentially try other l_p norms.

3 EXPERIMENTS

In this section, we mainly report results on three benchmark datasets: VOC2012 Everingham et al. (2014) and PASCAL-Context Mottaghi et al. (2014). VOC2012 has 20 object classes and one background class. Following Long et al. (2014); Chen et al. (2014), we augment it with extra annotations from Hariharan et al. (2011) that leads to 10,582, 1,449, and 1,456 images for training, validation, and testing. PASCAL-Context Mottaghi et al. (2014) fully labeled all scene classes appeared in VOC2010. We follow the same training + validation split as defined and used in Mottaghi et al. (2014); Long et al. (2014), resulting in 59 object + stuff classes and one background classes with 4,998 and 5105 training and validation images. All the results we describe below use the training images to train, and most of the results are on the validation set. We also report results on VOC2012 test set. We use Caffe Jia (2013) and fine-tune ParseNet from VGG-16 network Simonyan & Zisserman (2014) for different dataset.

3.1 BEST PRACTICE OF FINE-TUNING

As we know hyper-parameters are important for training/fine-tuning network, we try to reproduce the state-of-the-art systems' results by exploring the hyper-parameter space and achieve better baseline performance.

PASCAL-Context We start from the public system FCN-32s PASCAL-Context. Notice that it uses the accumulated gradient and affine transformation tricks that were introduced in Long et al. (2014). As such, it can deal with any input image of various sizes without warping or cropping it to fixed size, which can distort the image and affect the final segmentation result. Table 1 shows our different versions of reproduced baseline results. Baseline A uses the exactly same protocol, and our result is 1.5% lower. In Baseline B, we tried more iteration (160k vs. 80k) of fine-tuning and achieved similar performance to the reported one. Then, we modified the network a bit, i.e. we used "xavier" initialization Glorot & Bengio (2010), higher base learning rate (1e-9 vs. 1e-10), and lower momentum (0.9 vs. 0.99), and we achieved 1% higher accuracy as shown in Baseline C. What's more, we also remove the 100 padding in the first convolution layer and observed no significant difference but network trained slightly faster. Furthermore, we also used "poly" learning rate policy (base_lr $\times\ (1 - \frac{\text{iter}}{\text{max\_iter}})^{\text{power}}$, where power is set to 0.9.) as it is proved to converge faster than normal "step" policy, and thus can achieve 1.5% better performance with the same iterations (80k). All experimental results on PASCAL-Context are shown in table 1.

PASCAL-Context	Mean IoU
FCN-32s [3]	35.1
Baseline A	33.57
Baseline B	35.04
Baseline C	36.16
Baseline D	**36.64**

Table 1: **Reproduce FCN-32s on PASCAL-Context.** There are various modifications of the architecture that are described in Section 4.1.

PASCAL VOC2012 We carry over the hyper-parameters we found on PASCAL-Context to VOC2012. We tried both DeepLab and DeepLab-LargeFOV[4]. Table 2 shows the reproduced baseline results. DeepLab is very similar to FCN-32s, and our reproduced result, namely DeepLab Baseline, is 5.2% better (64.96 vs. 59.80) using the parameters we found in PASCAL-Context. DeepLab-LargeFOV uses the filter rarefication technique (atrous algorithm) that has much less parameters and is faster. We also use the same parameters on this architecture and can achieve 3.5% improvements. We name it as DeepLab-LargeFOV Baseline. The gap between these two models is not significant anymore as reported in Chen et al. (2014).

Until now, we see that parameters and details are important to get best performance using FCN models. Below, we report all our results with the reproduced baseline networks.

[3] https://gist.github.com/shelhamer/80667189b218ad570e82#file-readme-md

[4] https://bitbucket.org/deeplab/deeplab-public/

VOC2012	Mean IoU
DeepLab Chen et al. (2014)	59.80
DeepLab-LargeFOV Chen et al. (2014)	62.25
DeepLab Baseline	64.96
DeepLab-LargeFOV Baseline	**65.82**

Table 2: **Reproduce DeepLab and DeepLab-LargeFOV on PASCAL VOC2012.**

3.2 Combining Local and Global Features

In this section, we report results of combining global and local feature on three dataset: SiftFlow Liu et al. (2011), PASCAL-Context, and PASCAL VOC2012. For simplicity, we use pool6 as the global context feature, conv5 as conv5_3, conv4 as conv4_3, and conv3 as conv3_3 through the rest of paper.

SiftFlow is a relatively small dataset that only has 2,688 images with 33 semantic categories densely labeled. We do not use the geometric categories during training. We use the DeepLab Baseline network with the hyper-parameters found in PASCAL-Context. Instead of using two stages of learning as done in Long et al. (2014), we combine the feature directly from different layers for learning. As shown in Table 3, adding more layers can normally improve the performance as lower level layers have more detailed information. We also notice that adding global context feature does not help much. We hypothesize that global context is less helpful for small images.

	pixel	mean	mean	f.w.
	acc.	acc.	IU	IU
FCN-16s Long et al. (2014)	85.2	51.7	39.5	76.1
fc7	85.1	44.1	35.4	75.6
pool6 + fc7	85.7	43.9	35.5	76.4
pool6 + fc7 + conv5	85.4	51.4	38.7	76.3
pool6 + fc7 + conv5 + conv4	**86.8**	**52.0**	**40.4**	**78.1**

Table 3: **Results on SiftFlow.** Early fusion can work equally well as late fusion as used in Long et al. (2014). Adding more layers of feature generally increase the performance. Global feature is not that helpful as receptive field size of fc7 is large enough to cover most of the input image.

PASCAL-Context We then apply the DeepLab Baseline model on PASCAL-Context by concatenating features from different layers of the network. As shown in Table 4, by adding global context pool6, it instantly helps improve by about 1.6%, which means that context is useful here as opposed to the observation in SiftFlow. Context becomes more important proportionally to the image size. Another interesting observation from the table is that, without normalization, the performance keep increasing until we add conv5. However, if we naively keep adding conv4, it starts decreasing the performance a bit; and if we add conv3, the network collapses. Interestingly, if we normalize all the features before we combine them, we don't see such a drop, instead, adding all the feature together can achieve the state-of-the-art result on PASCAL-Context as far as we know.

	w/o Norm	w/ Norm
FCN-32s	35.1	N/A
FCN-8s	37.8	N/A
fc7	36.6	36.2
pool6 + fc7	38.2	37.6
pool6 + fc7 + conv5	39.5	39.9
pool6 + fc7 + conv5 + conv4	36.5	40.2
pool6 + fc7 + conv5 + conv4 + conv3	0.009	**40.4**

Table 4: **Results on PASCAL-Context.** Adding more layers helps if we L_2 normalize them.

PASCAL VOC2012 Since we have reproduced both DeepLab Baseline and DeepLab-LargeFOV Baseline on VOC2012, we want to see how does global context, normalization, and early or late fusion affect performance.

We start with using DeepLab Baseline, and try to add pool6 to it. It improves from 64.92% to 67.49% by adding pool6 with normalization. Interestingly, without normalizing fc7 and pool6, we don't see any improvements. As opposed to what we observed from SiftFlow and PASCAL-Context. We hypothesize this is due to images in VOC2012 mostly have one or two objects in the image versus the other two dataset who have multiple labels per image, and we need to adjust the weight more carefully to make the context feature more useful. It also suggests that context is more useful for sparsely labeled images.

DeepLab-LargeFOV Baseline performance is higher than DeepLab Baseline and it is faster, thus we switch to use it for most of the experimental comparison for VOC2012. As shown in Table 5, we observe a similar pattern as of DeepLab Baseline that if we add pool6, it is helping improve the performance by 3.8%. However, we also notice that if we do not normalize them and learn the scaling factors, its effect is diminished. Furthermore, we notice that early fusion and late fusion both work very similar. Figure 4 illustrates some examples of how global context helps. We can clearly see that without using context feature, the network will make many mistakes by confusing between similar categories as well as making spurious predictions. Two similar looking patches are indistinguishable by the network if considered in isolation. However, adding context solves this issue as the global context helps discriminate the local patches more accurately. On the other hand, sometimes context also brings confusion for prediction as shown in Figure 5. For example, in the first row, the global context feature definitely captured the spotty dog information that it used to help discriminate sheep from dog. However, it also added bias to classify the spotty horse as a dog. The other three examples have the same issue. Overall, by learning to weight pool6 and fc7 after L_2 normalization helps improve the performance greatly.

Layers	Norm (Y/N)	Early or Late (E/L)	Mean IoU
fc7	N	NA	65.82
fc7	Y	NA	65.66
pool6 + fc7	N	E	65.30
pool6 + fc7	Y	E	69.43
pool6 + fc7	Y	L	**69.55**
pool6 + fc7	N	L	69.29

Table 5: **Add context for DeepLab-LargeFOV Baseline on VOC2012.**

We also tried to combine lower level feature as was done with PASCAL-Context and SiftFlow, but no significant improvements using either *early fusion* or *late fusion* were observed. We believe it is because the fc7 of DeepLab-LargeFOV Baseline is the same size as of conv4, and including lower level feature will not help much as they are not sufficiently discriminative. Besides, we also tried the idea similar to spatial pyramid pooling where we pool 1×1 global feature, 2×2 subregion feature, and 4×4 subregion feature, and tried both *early fusion* and *late fusion*. However, we observed no improvements. We conjecture that the receptive field of the high level feature map (e.g. fc7) is sufficiently large that sub-region global feature does not help much.

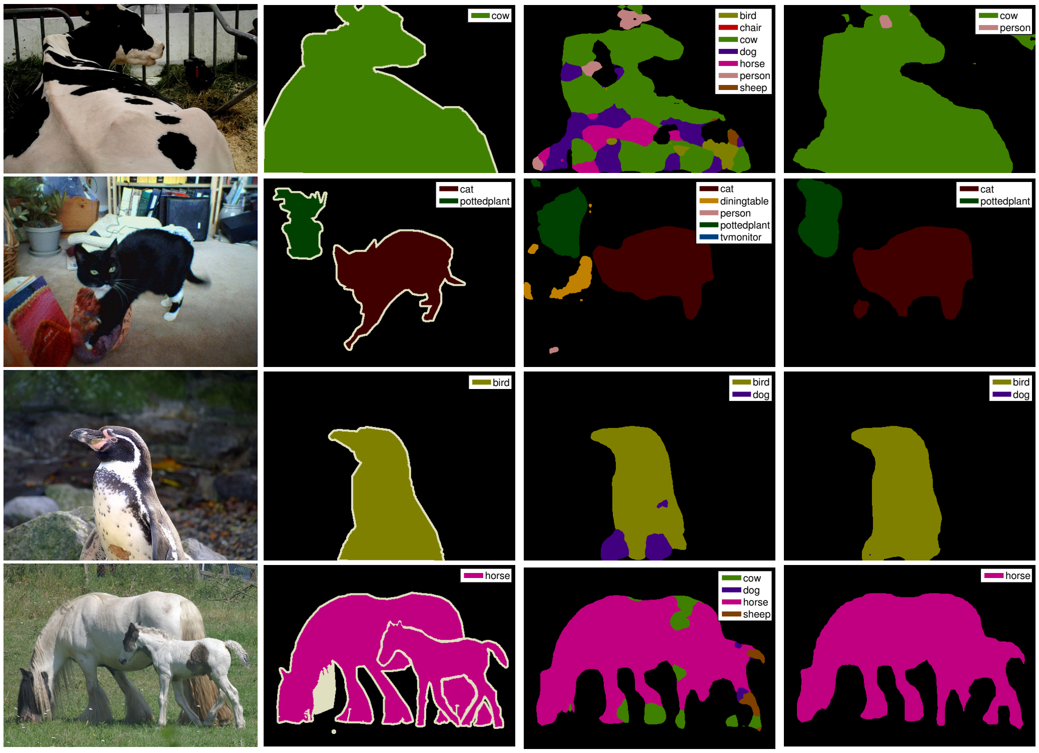
System	**bkg**	**aero**	**bike**	**bird**	**boat**	**bottle**	**bus**	**car**	**cat**	**chair**	**cow**	**table**	**dog**	**horse**	**mbike**	**person**	**plant**	**sheep**	**sofa**	**train**	**tv**	*mean*
FCN-8s	-	76.8	34.2	68.9	49.4	60.3	75.3	74.7	77.6	21.4	62.5	46.8	71.8	63.9	76.5	73.9	45.2	72.4	37.4	70.9	55.1	62.2
Hypercolumn	-	68.7	33.5	69.8	51.3	70.2	81.1	71.9	74.9	23.9	60.6	46.9	72.1	68.3	74.5	72.9	52.6	64.4	45.4	64.9	57.4	62.6
TTI-Zoomout-16	89.8	81.9	35.1	78.2	57.4	56.5	80.5	74.0	79.8	22.4	69.6	53.7	74.0	76.0	76.6	68.8	44.3	70.2	40.2	68.9	55.3	64.4
DeepLab-CRF-LargeFOV	**92.6**	83.5	36.6	**82.5**	62.3	**66.5**	85.4	78.5	83.7	**30.4**	72.9	**60.4**	**78.5**	75.5	**82.1**	**79.7**	**58.2**	**82.0**	48.8	73.7	63.3	**70.3**
DeepLab-LargeFOV Baseline [5]	92.3	82.6	36.1	76.1	59.3	62.3	81.6	79.5	81.4	28.1	70.0	53.0	73.2	70.6	78.8	78.6	51.9	77.4	45.5	71.7	62.6	67.3
ParseNet [6]	92.4	**84.1**	**37.0**	77.0	**62.8**	64.0	**85.8**	**79.7**	**83.7**	27.7	**74.8**	57.6	77.1	**78.3**	81.0	78.2	52.6	80.4	**49.9**	**75.7**	**65.0**	69.8

Table 6: PASCAL VOC2012 test Segmentation results.

Finally, we test two models, DeepLab-LargeFOV Baseline and ParseNet, on VOC2012 test set. ParseNet is DeepLab-LargeFOV Baseline plus global context. As shown in Table 6, we can see

[5] `http://host.robots.ox.ac.uk:8080/anonymous/LGOLRG.html`

[6] `http://host.robots.ox.ac.uk:8080/anonymous/56QLXU.html`



(a) Original Image (b) Ground truth (c) DeepLab-LargeFOV Baseline (d) ParseNet

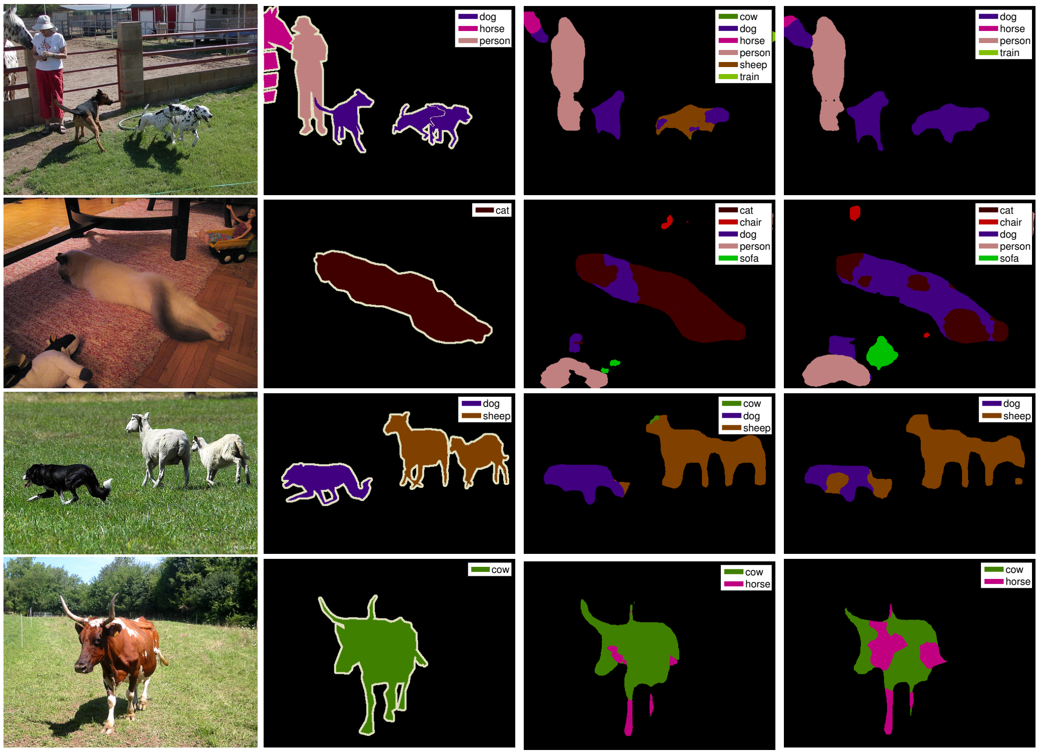
Figure 4: Global context helps for classifying local patches.

that our baseline result is already higher than many existing methods due to proper fine-tuning. By adding the global context feature, we achieve performance that is within the standard deviation of the one Chen et al. (2014) using fully connect CRF (DeepLab-CRF-LargeFOV) to smooth the outputs and perform better on more than half of categories. Again, our approach is simple to implement and train, hence is more robust. Using *late fusion* has almost no extra training/inference cost.

4 Related Work

Deep convolutional neural networks (CNN) Krizhevsky et al. (2012); Szegedy et al. (2014a); Simonyan & Zisserman (2014) have become powerful tools not only for whole image classification, but also for object detection and semantic segmentation Girshick et al. (2014); Szegedy et al. (2014b); Gupta et al. (2014). This success has been attributed to both the large capacity and effective training of the CNN. Following the *proposal + post-classification* scheme Uijlings et al. (2013), CNNs achieve state-of-the-art results on object detection and segmentation tasks. As a caveat, even though a single pass through the networks used in these systems is approaching or already past video frame rate for individual patch, these approaches require classifying hundreds or thousands of patches per image, and thus are still slow. He et al. (2014); Long et al. (2014) improve the computation by applying convolution to the whole image once, and then pool features from the final feature map of the network for each region proposal or pixel to achieve comparable or even better results. Yet, these methods still fall short of including whole image context and only classify patches or pixels locally. Our ParseNet is built upon the fully convolutional network architecture Long et al. (2014) with a strong emphasis on including contextual information in a simple approach.

For semantic segmentation, using context information Rabinovich et al. (2007); Shotton et al. (2009); Torralba (2003); Gonfaus et al. (2010) from the whole image can significantly help classifying local patches. Lucchi et al. (2011) shows that by concatenating features from the whole image to the local patch, the inclusion of post processing (i.e. CRF smoothing) becomes unnecessary because the image level features already encode the smoothness. Mostajabi et al. (2014) demonstrate that



(a) Original Image (b) Ground truth (c) DeepLab-LargeFOV Baseline (d) ParseNet

Figure 5: Global context confuse local patch predictions.

by using the ”zoom-out” features, which is a combination of features for each super pixel, region surrounding it, and the whole image, they can achieve impressive performance for the semantic segmentation task. These approaches pool features differently for local patches and the whole image, making it difficult to train the whole system end-to-end. Exploiting the FCN architecture, ParsetNet can directly use global average pooling from the final (or any) feature map, resulting in the feature of the whole image, and use it as context. Experiments results confirm that ParseNet can capture the context of the image and thus improve local patch prediction results.

There is another line of work that attempts to combine graphical models with CNNs to incorporate both context and smoothness priors. Chen et al. (2014) first uses a FCN to estimate the unary potential, then applies a fully connected CRF to smooth the predictions spatially. As this approach consists of two decoupled stages, it is difficult to train the FCN properly to minimize the final objective of smooth and accurate semantic segments. A more unified and principled approach is to incorporate the structure information during training directly. Schwing & Urtasun (2015) propagates the marginals computed from the structured loss to update the network parameters, Lin et al. (2015) uses piece-wise training to make learning more efficient by adding a few extra piece-wise networks, while Zheng et al. (2015) convert CRF learning to recurrent neural network (RNN) and use message passing to do the learning and inference. However, we show that our method can achieve comparable accuracy, with a simpler – hence more robust – structure, while requiring only a small amount of additional training/inference time.

5 Conclusion

In this work we presented ParseNet, a simple fully convolutional neural network architecture that allows for direct inclusion of global context for the task of semantic segmentation. We have explicitly demonstrated that relying on the largest receptive field of FCN network does not provide sufficient global context, and the largest empirical receptive field is not sufficient to capture global context – modeling global context directly in required. On PASCAL VOC2012 test set, segmentation results of ParseNet are within the standard deviation of the DeepLab-LargeFOV-CRF, which suggests that adding a global feature has a similar effect of post processing FCN predictions with a graphical model. As part of developing and analyzing this approach we provided analysis of many architectural choices for the network, discussing best practices for training, and demonstrated the importance of normalization and learning weights when combining features from multiple layers of a network. By themselves, our practices for training significantly improve the baselines we use before adding global context. The guiding principle in the design of ParseNet is simplicity and robustness of learning. Results are presented on three benchmark dataset, and are state of the art on SiftFlow and PASCAL-Context, and near the state of the art on PASCAL VOC2012. Given the simplicity and ease of training, we find these results very encouraging. In our on going work, we are exploring combining our technique with structure training/inference as done in Schwing & Urtasun (2015); Lin et al. (2015); Zheng et al. (2015).

REFERENCES

Liang-Chieh Chen, George Papandreou, Iasonas Kokkinos, Kevin Murphy, and Alan L Yuille. Semantic image segmentation with deep convolutional nets and fully connected crfs. *arXiv:1412.7062*, 2014.

Mark Everingham, SM Ali Eslami, Luc Van Gool, Christopher KI Williams, John Winn, and Andrew Zisserman. The pascal visual object classes challenge: A retrospective. *IJCV*, 2014.

Ross Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *CVPR*, 2014.

Xavier Glorot and Yoshua Bengio. Understanding the difficulty of training deep feedforward neural networks. In *International conference on artificial intelligence and statistics*, 2010.

Josep M Gonfaus, Xavier Boix, Joost Van de Weijer, Andrew D Bagdanov, Joan Serrat, and Jordi Gonzalez. Harmony potentials for joint classification and segmentation. In *CVPR*, 2010.

Saurabh Gupta, Ross Girshick, Pablo Arbeláez, and Jitendra Malik. Learning rich features from rgb-d images for object detection and segmentation. In *ECCV*, 2014.

Bharath Hariharan, Pablo Arbeláez, Lubomir Bourdev, Subhransu Maji, and Jitendra Malik. Semantic contours from inverse detectors. In *ICCV*, 2011.

Bharath Hariharan, Pablo Arbeláez, Ross Girshick, and Jitendra Malik. Hypercolumns for object segmentation and fine-grained localization. *arXiv:1411.5752*, 2014.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Spatial pyramid pooling in deep convolutional networks for visual recognition. *arXiv:1406.4729*, 2014.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. *arXiv:1502.01852*, 2015.

Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *arXiv:1502.03167*, 2015.

Yangqing Jia. Caffe: An open source convolutional architecture for fast feature embedding. `http://caffe.berkeleyvision.org`, 2013.

Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *NIPS*, 2012.

Svetlana Lazebnik, Cordelia Schmid, and Jean Ponce. Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories. In *CVPR*, 2006.

Guosheng Lin, Chunhua Shen, Ian Reid, et al. Efficient piecewise training of deep structured models for semantic segmentation. *arXiv:1504.01013*, 2015.

Ce Liu, Jenny Yuen, and Antonio Torralba. Nonparametric scene parsing via label transfer. *PAMI*, 2011.

Jonathan Long, Evan Shelhamer, and Trevor Darrell. Fully convolutional networks for semantic segmentation. *arXiv:1411.4038*, 2014.

Aurelien Lucchi, Yunpeng Li, Xavier Boix, Kevin Smith, and Pascal Fua. Are spatial and global constraints really necessary for segmentation? In *ICCV*, 2011.

Mohammadreza Mostajabi, Payman Yadollahpour, and Gregory Shakhnarovich. Feedforward semantic segmentation with zoom-out features. *arXiv:1412.0774*, 2014.

Roozbeh Mottaghi, Xianjie Chen, Xiaobai Liu, Nam-Gyu Cho, Seong-Whan Lee, Sanja Fidler, Raquel Urtasun, and Alan Yuille. The role of context for object detection and semantic segmentation in the wild. In *CVPR*, 2014.

Andrew Rabinovich, Andrea Vedaldi, Carolina Galleguillos, Eric Wiewiora, and Serge Belongie. Objects in context. In *CVPR*, 2007.

Alexander G Schwing and Raquel Urtasun. Fully connected deep structured networks. *arXiv:1503.02351*, 2015.

Jamie Shotton, John Winn, Carsten Rother, and Antonio Criminisi. Textonboost for image understanding: Multi-class object recognition and segmentation by jointly modeling texture, layout, and context. *IJCV*, 2009.

Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv:1409.1556*, 2014.

Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. *arXiv:1409.4842*, 2014a.

Christian Szegedy, Scott Reed, Dumitru Erhan, and Dragomir Anguelov. Scalable, high-quality object detection. *arXiv:1412.1441*, 2014b.

Antonio Torralba. Contextual priming for object detection. *IJCV*, 2003.

Jasper RR Uijlings, Koen EA van de Sande, Theo Gevers, and Arnold WM Smeulders. Selective search for object recognition. *IJCV*, 2013.

Shuai Zheng, Sadeep Jayasumana, Bernardino Romera-Paredes, Vibhav Vineet, Zhizhong Su, Dalong Du, Chang Huang, and Philip Torr. Conditional random fields as recurrent neural networks. *arXiv:1502.03240*, 2015.

Bolei Zhou, Aditya Khosla, Agata Lapedriza, Aude Oliva, and Antonio Torralba. Object detectors emerge in deep scene cnns. *arXiv preprint arXiv:1412.6856*, 2014.