Dense Decoder Shortcut Connections for Single-Pass Semantic Segmentation

Piotr Bilinski
University of Oxford
piotrb@robots.ox.ac.uk

Victor Prisacariu
University of Oxford
victor@robots.ox.ac.uk

Abstract

We propose a novel end-to-end trainable, deep, encoder-decoder architecture for single-pass semantic segmentation. Our approach is based on a cascaded architecture with feature-level long-range skip connections. The encoder incorporates the structure of ResNeXt's residual building blocks and adopts the strategy of repeating a building block that aggregates a set of transformations with the same topology. The decoder features a novel architecture, consisting of blocks, that (i) capture context information, (ii) generate semantic features, and (iii) enable fusion between different output resolutions. Crucially, we introduce dense decoder shortcut connections to allow decoder blocks to use semantic feature maps from all previous decoder levels, i.e. *from all higher-level feature maps. The dense decoder connections allow for effective information propagation from one decoder block to another, as well as for multi-level feature fusion that significantly improves the accuracy. Importantly, these connections allow our method to obtain state-of-the-art performance on several challenging datasets, without the need of time-consuming multi-scale averaging of previous works.*

1. Introduction

Semantic segmentation is the process of assigning a class label to each pixel in an image, which is pertinent in a number of applications including scene understanding for robotic vision, autonomous driving, localization, as well as navigation.

Recent works on semantic segmentation have shown significant improvements in accuracy by incorporating state-of-the-art deep Convolutional Neural Networks (CNN) based image classifiers for semantic segmentation [5, 36]. One prominent approach involves Fully Convolutional Networks (FCN) [36], where the fully-connected layers are converted into convolutional ones. We note that crucial in this approach is the use of a skip architecture, which combines semantic information from deep, coarse layers with appearance information from shallow, fine layers.

Most recent techniques have incorporated image pyramids for multi-scale inference and evaluation [5, 7, 9, 26, 31, 39, 54]. All such techniques take an input image and scale it to multiple resolutions to build a set of cascading images, which are then piped, most often independently, through a (neural network) processing pipeline. This technique provides useful properties for many applications, such as noise reduction or image manipulation. It is widely used in semantic segmentation as an image representation for extracting multi-scale semantic scores, which are then averaged across all the scales to produce a final result. However, running the inference step for many scales separately significantly increases the processing time, thus making such an approach impractical for many real-world applications. More importantly, the multi-scale pyramid significantly increases memory requirements, making it virtually impossible to train a network end-to-end using a single, modern GPU. This indicates that the multi-scale image pyramid is used for the inference step only, often making the whole approach heavily-engineered.

Motivated by the above, in this paper we design a novel end-to-end trainable deep encoder-decoder architecture, that aims to eliminate the need for multi-scale, multi-pass semantic segmentation, and achieves comparable or better results using a *single* scale. Our approach is based on the cascaded architecture with feature-level long-range skip connections. The encoder incorporates the structure of ResNeXt's residual building blocks and adopts the strategy of repeating a building block that aggregates a set of transformations with the same topology. The decoder features a novel architecture consisting of blocks, that capture context information, generate semantic features and enable fusion between different resolution levels. Crucially, we propose to use dense decoder shortcut connections to allow decoder's blocks to use semantic feature maps from all previous decoder's blocks (*i.e.* from all higher-level feature maps) and correct any potential errors introduced by the previous decoder's blocks. The dense decoder shortcut connections allow for effective information propagation from one block of a decoder to another, and for multi-level feature fusion that significantly improves the accuracy.

We perform an extensive evaluation involving several commonly used semantic segmentation datasets: Pascal VOC 2012 [13], Pascal-Context [38], Pascal Person-Part [8], NYUD [47] and CamVid [3]. Our experiments show that the proposed approach achieves single-scale state-of-the-art results, often outperforming even multi-scale methods.

2. Related Work

Semantic Segmentation. Early works on semantic segmentation have focused on designing a strong feature representation, *e.g.* TextonBoost [46], TextonForest [44], as well as Random Forest-based classifiers [45]. Recently, supervised deep learning architectures have observed widespread success in many areas and have also become the standard for successful approaches for semantic segmentation. In particular, a number of methods, including DeepLab [5] and Fully Convolutional Networks (FCN) [36], have shown significant improvements in accuracy by adapting state-of-the-art deep Convolutional Neural Networks (CNN) based image classifiers to semantic segmentation. The DeepLab [5, 6] investigates atrous (*i.e.* dilated) convolutions, which allow to control the resolution at which feature responses are computed, and effectively enlarge the field of view of filters to incorporate larger context without increasing the number of parameters or the amount of computation. The DeepLab is based on the VGG-16 [48], where the fully-connected layers have been converted into convolutional ones. Moreover, the responses at the final CNN layer have been combined with a fully connected Conditional Random Fields (CRF), that has been applied at the post-processing stage. The FCN approach [36], has also adapted the VGG-16 architecture, but into an end-to-end trainable fully convolutional network. Moreover, FCN has defined a skip architecture that combines semantic information from deep, coarse layers with appearance information from shallow, fine layers. This method has been the basis for many further works [1, 35, 53, 55].

The aforementioned deep learning techniques are mainly based on the VGG-16 architecture [48]. Recent research works have shown that substantially deeper architectures, such as deep residual networks ResNets [18] and ResNeXts [52] (ResNet-style layers, but with split-transform-merge strategy), can gain accuracy from considerably increasing the depth of the network. These residual networks (i) are substantially deeper (*e.g.* with 50, 101, or 152 layers) than those used in the past (*e.g.* ResNet with 152 layers has 9.5 times more layers than the VGG-16 [48]), (ii) have fewer parameters than the VGG nets, (iii) are easier to optimize, and (iv) can gain accuracy from considerably increased depth [52]. PSPNet [54] and RefineNet [31] are a few of the recent methods that have applied the ResNet architecture in semantic segmentation. The former also adopts a region-based context aggregation through a pyramid pooling module. The latter uses a cascaded architecture. In some respects, this is the closest work to ours, but there are many important differences between our methods: (i) RefineNet incorporates image pyramids for multi-scale inference and evaluation, whereas we propose dense decoder shortcut connections for single-pass inference and evaluation, (ii) both methods propose different decoders, (iii) both methods aggregate context in different ways, and (iv) both methods incorporate different building blocks. Moreover, our approach significantly outperforms the RefineNet, even up to 4.5% on the NYUD dataset.

CRF. The Conditional Random Fields (CRF) is a commonly used graphical model for image segmentation. It incorporates smoothness terms that maximize label agreement between similar pixels. Many recent works have explored joint learning of the FCN and CRF; on fully integrating the CRF modeling with CNN, leveraging on training of the whole network end-to-end [55, 1]. In [55], the FCN was used with the mean-field approximate inference formulation for CRF with Gaussian pairwise potentials as Recurrent Neural Networks. In [1], higher order potentials (object detection and super-pixel based potentials) were incorporated into an end-to-end trainable CRF model. **Object Proposals.** There are many other techniques that refine bounding boxes of objects [16, 17] or segment-based region proposals [40, 41] into a segmentation. Although these methods (*i.e.* CRF and object-based/region-based proposals) can be incorporated in our network in order to further boost the approach accuracy, these methods are out of the scope of this paper.

Dense Connections. The idea of dense connections has been recently proposed for image classification in DenseNet [20] and extended to semantic segmentation in FC-DenseNet [24]. However, there are many important differences in the use and purpose of dense connections between our methods. The above methods use multiple dense blocks with transition layers between these blocks (there are no dense connections between dense blocks), whereas we use dense connections between decoder's blocks, which are on a different level of abstraction, see Fig. 1. Moreover, the above methods use dense connections to create more efficient building blocks, whereas we use them to strengthen feature propagation for multi-level semantic feature fusion. Furthermore, our approach significantly outperforms the FC-DenseNet, even up to 4% on the CamVid dataset.

3. Proposed Method

Structure. Our semantic segmentation network consists of 2 main parts: an encoder and a decoder, see Fig. 1. The encoder consists of 4 blocks, named: enc_1, enc_2, enc_3 and enc_4. The encoder extracts appearance information on var-

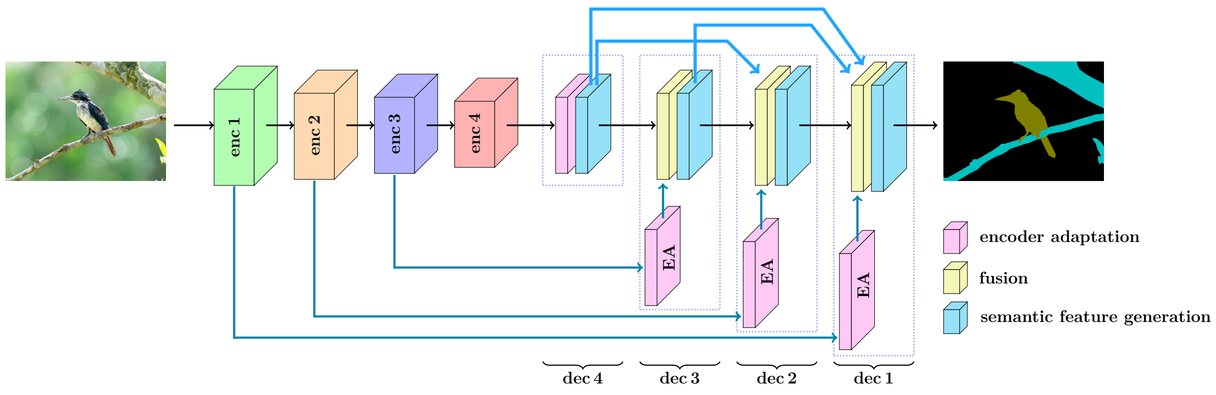

Figure 1: Overview of our architecture for single-pass semantic segmentation: cascaded architecture with our context-aggregating decoder, feature-level long-range skip connections, and dense decoder shortcut connections for multi-level feature fusion.

ious layers, from shallow, fine layers (enc_1) to deep, coarse layers (enc_4). The decoder consists of 4 main blocks, which we call: dec_4, dec_3, dec_2 and dec_1. It provides semantic segmentation results on various hierarchical levels, from coarse (dec_4) to fine (dec_1) segmentations.

Our network is built upon a cascaded architecture. The output of the block enc_x, where $x = 1, \dots, 3$, is connected to the input of the block enc_{x+1}, the output of the block enc_4 is connected to the input of the block dec_4, and each output of the block dec_x, where $x = 2, \dots, 4$, is connected to the input of the block dec_{x-1}. Moreover, each decoder's blocks dec_x, where $x = 1, \dots, 3$, uses a feature map from the corresponding encoder's block enc_x to refine a low resolution feature map from the previous decoder's block dec_{x+1}.

Encoder. We base the architecture of our encoder on residual building blocks of the ResNeXt, but we remove the global average pooling, fully connected layer and softmax from the network. The architecture of our encoder, *i.e.* blocks enc_x, where $x = 1, \dots, 4$, based on the ResNeXt-101 are presented in Tab. 1.

Decoder. We propose a new architecture of a semantic segmentation decoder. The core of the decoder consists of 4 blocks, *i.e.* dec_x, where $x = 1, \dots, 4$. An overview of the decoder's blocks is presented in Fig. 1. Each one of the blocks consists of an encoder adaptation stage and a semantic feature generation stage. Moreover, the blocks dec_1, dec_2, and dec_3 have a fusion stage in between them. We call these 3 blocks the fusion blocks, as they fuse the output of the previous decoder with the output from the encoder adaptation. The block dec_4 has only one input and therefore, it does not perform any fusion. Note that all of the convolution and pooling operations in the decoder, explained below, do not change the dimensionality of the features, unless stated otherwise.

The **encoder adaptation** is the first stage of every decoder's block dec_x, where $x = 1, \dots, 4$, see Fig. 1. Its

stage	template	
	7×7, 64, stride 2	
	3×3 max pool, stride 2	
enc1	$\begin{bmatrix} 1 \times 1,\ 128 \\ 3 \times 3,\ 128,\ C{=}32 \\ 1 \times 1,\ 256 \end{bmatrix}$	$\times 3$
enc2	$\begin{bmatrix} 1 \times 1,\ 256 \\ 3 \times 3,\ 256,\ C{=}32 \\ 1 \times 1,\ 512 \end{bmatrix}$	$\times 4$
enc3	$\begin{bmatrix} 1 \times 1,\ 512 \\ 3 \times 3,\ 512,\ C{=}32 \\ 1 \times 1,\ 1024 \end{bmatrix}$	$\times 23$
enc4	$\begin{bmatrix} 1 \times 1,\ 1024 \\ 3 \times 3,\ 1024,\ C{=}32 \\ 1 \times 1,\ 2048 \end{bmatrix}$	$\times 3$
EA	$3 \times 3,\ D$	
	$\begin{bmatrix} 1 \times 1,\ D/2 \\ 3 \times 3,\ D/2,\ C{=}32 \\ 1 \times 1,\ D \end{bmatrix}$	$\times 3$
	$3 \times 3,\ D$	

Table 1: Architecture of our encoder's blocks enc_x, where $x{=}1, \dots, 4$, and our encoder adaptation (EA) stage. The shape of the residual building blocks are shown in brackets, with the number of blocks stacked for each of the stages. A layer is denoted as (filter size, # output channels). "C=32" refers to grouped convolutions with 32 groups.

purpose is to prepare features from an encoder enc_x for the fusion stage (for $x = 1, \dots, 3$) or semantic feature generation stage (for $x = 4$). Firstly, we apply a 3×3 convolution to reduce the number of channels of features enc_x, as these features are typically of high dimension (*e.g.* 2048 for the enc_4). Then, we apply three stacked residual building blocks to adapt features for the consecutive stages, see

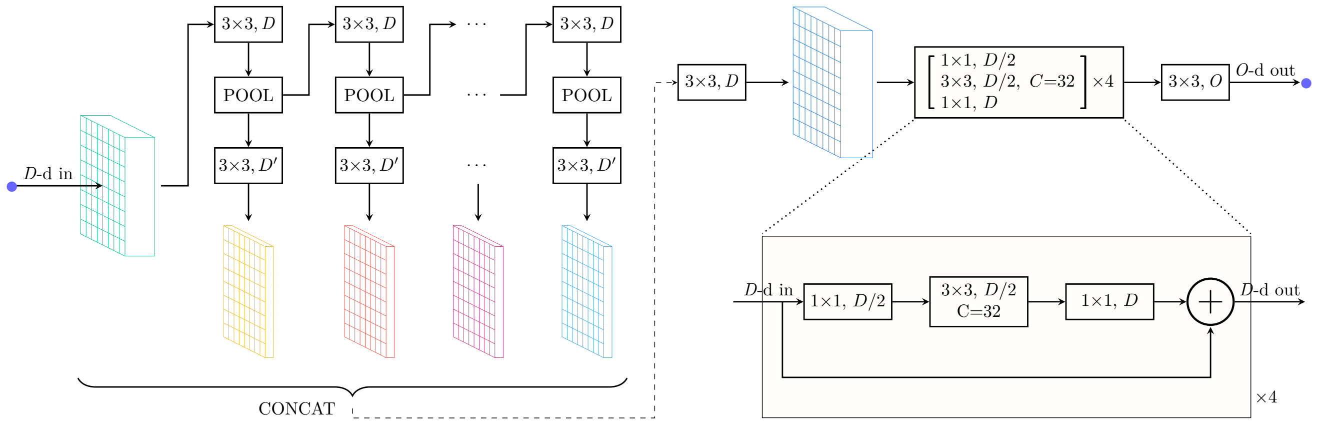

Figure 2: Decoder's semantic feature generation stage.

Tab. 1. Finally, if a decoder's block performs a fusion (*i.e.* dec_1, dec_2 and dec_3), we also apply a 3×3 convolution (see the explanation below).

The **fusion** is the second stage of the blocks dec_1, dec_2 and dec_3. The purpose of this stage is to fuse features from the encoder adaptation with features from the previous decoders. In particular, in decoder dec_x, we use features enc_x, that passed through the encoder adaptation stage, to refine low resolution features from the previous decoders $\text{dec}_{x+1,..,4}$. However, we must first ensure that the features we want to fuse have the same dimensionality, *i.e.* they have the same spatial resolution and number of channels. To ensure the same number of channels, the last operation of the encoder adaptation and the last operation of the previous decoder undergo a 3×3 convolution. Therefore, we simply set the number of output channels of both convolutions to the same value, *i.e.* the smallest value among the number of input channels of these convolutions. Then, to ensure the same spatial resolution of the features, we apply a bilinear interpolation to up-scale low-resolution features to the largest spatial resolution of the features that we want to fuse. Finally, we can fuse the features by element-wise summation operation, as the features have the same dimensionality, see Fig. 3.

The last section of each block in the decoder is responsible for capturing context information and then using it to **generate semantic features**. To capture context information, we apply a sequence of 4 convolution-pooling blocks, each with a 3×3 convolution followed by a 5×5 max pooling layer. Each pooling block captures context information from a large area in the image for every spatial position of the feature map. We fuse the input of this stage with all outputs of the pooling operations by concatenation of the feature maps. We also apply 3×3 convolutions to reduce (i) the dimension of the feature maps from the pooling layers, and (ii) the dimension of the concatenated features.

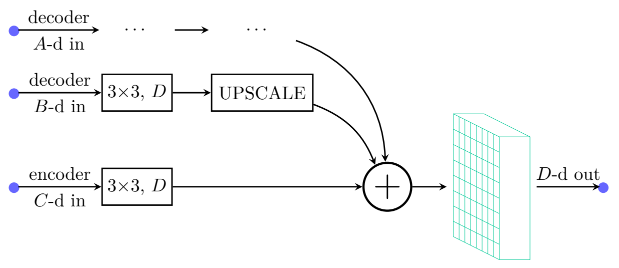

Figure 3: Decoder's fusion stage.

This is shown in Fig 2. We then generate semantic features for further processing (in blocks dec_4, dec_3 and dec_2) or final prediction (in the block dec_1) by applying four stacked residual building blocks. If we generate semantic features for further processing, we then apply a 3×3 convolution, as explained in the fusion stage. If we generate semantic features for final prediction, we apply a dropout to reduce overfitting followed by a 3×3 convolution to adjust the number of output channels of a feature map to the number of classes. Moreover, we apply per-pixel softmax function to generate a semantic segmentation map, and we also apply the bilinear interpolation to up-scale a low-resolution feature map to the original size of an image.

Image Pyramid. The image pyramid has been widely used in semantic segmentation as a multi-scale image representation for extracting multi-scale image features. Usually, after an image pyramid is created, semantic segmentation is run on each scale separately, and the results are averaged across the different scales [5, 7, 9, 26, 31, 39, 54]. However, running inference step separately for each scale significantly increases the processing time, making such an approach impractical for many real world applications. Moreover, using an image pyramid significantly increases the memory requirements, making it impossible to train the entire network end-to-end with a single GPU. Therefore, the image pyramid is only used for the inference step, making such approaches heavily-engineered.

Dense Decoder Shortcut Connections. Instead of using the image pyramid directly, we introduce the dense decoder shortcut connections, see Fig 1. We treat the decoder's blocks as a hierarchy of semantic features, as if it was created by the image pyramid. Our cascaded architecture effectively enables the creation of multi-scale feature pyramid from just a single-pass image inference. Our dense decoder shortcut connections use the concept of a feature pyramid, and connect every output of a decoder's block dec_x with every input of a decoder's block dec_y, where $x > y$. This is to allow the decoder's blocks to use semantic feature maps from all previous (higher-level) decoder's blocks to correct any potential errors previously introduced in the decoder. Our dense decoder shortcut connections effectively fuse the semantic feature maps from different scales, and allow for effective information propagation from one block of to another. This significantly reduces the memory requirements and allows us to train the entire network end-to-end on a single GPU with 12 GB of memory, whilst using the multi-scale feature pyramid. Moreover, the image pyramid for multi-scale inference and evaluation can still be used, if needed, resulting in pyramid of multi-scale feature pyramids.

Furthermore, we have 2 other types of shortcut connections in our network: short-range and long-range. The short-range skip connections are the identity mappings of residual building blocks, see Fig. 2, which allow the forward and backward signals to be directly propagated from one block to any other block [19]. The feature-level long-range skip connections combine semantic information from deep, coarse layers with appearance information from shallow, fine layers. We extend the idea of using 2 long-range skip connections from the FCN-8 [36] to 3 long-range shortcut connections, but with such difference, that the FCN's skip connections are applied to fuse predictions, whereas ours are applied to fuse semantic features. Our long-range skip connections occur between blocks enc_x and dec_x, where $x \in \{1, 2, 3\}$, and are shown in Fig. 1.

4. Experiments

We comprehensively evaluate our method on several commonly used semantic segmentation datasets: Pascal VOC 2012 [13], Pascal-Context [38], Pascal Person-Part [8], NYUD [47] and CamVid [3]. In the following subsections, we present experimental setup, results and comparison with the state-of-the-art.

Experimental Setup. We train the entire network end-to-end with the usual back-propagation algorithm using only a single NVIDIA GeForce GTX TITAN X with 12 GB of memory. We train our network with common and simple data augmentation strategies such as random scaling (from 0.7 to 1.3), horizontal flipping and random cropping. For the encoder, we use the architecture of the ResNeXt-101; however, any of the ResNet and ResNeXt architectures could be used. We initialize the layer weights of our encoder using those from the ResNeXt model, which was pre-trained on ImageNet [43], and then we fine-tune in turn the finer strides on the training/validation data. The batch normalization [22, 52] is applied in the encoder only, and we fix it after the network is pre-trained, because the batch size is too small to build up meaningful batch statistics. For the decoder, we use the ReLU activation function right after each convolution and element-wise summation operation, except for the output of the block, where ReLU is performed after adding to the shortcut, following the concepts of [18, 52]. Moreover, we do not use the ReLU right after the last convolution of the encoder adaptation stage and semantic feature generation stage, where features are being adopted for the fusion stage or final prediction, and before the pooling layer in the semantic feature generation stage. For fine-tuning, we set the learning rate to 1e-4, the weight decay to 5e-4, and the momentum to 0.9. Moreover, we set the decoder's parameters as follows: D=1024 for dec_4 and D=#input channels of encoder's features for the remaining blocks, and D'=$D/4$. We have also experimented with various lengths of convolution-pooling blocks and various max pooling windows. The selected parameters have been empirically shown to give a good trade-off between convergence time and results. To up-scale features, we have considered both a bilinear interpolation and a transposed convolution with learnable weights; however, both methods have shown to achieve similar accuracy, and thus we have selected a simpler bilinear interpolation as it requires fewer parameters to learn. We report the mean IoU as the commonly used semantic segmentation metric [13, 36].

The **NYUD** (NYU Depth v2) dataset [47] consists of 1449 RGB and depth images (381 training, 414 validation and 654 testing). We train our approach on RGB images only, without using the depth information, and we follow the commonly used 40 classes task [14, 36]. Our results are presented in Tab. 2. Sample predictions of our approach are shown in Fig. 5. Our approach using a single-scale evaluation achieves a new state-of-the-art accuracy 48.1 mean IoU, significantly outperforming the state-of-the-art methods using a multi-scale evaluation.

Then, we use the NYUD dataset for various experiments due to the small number of images. Firstly, we show that our dense decoder shortcut connections (DDSC) improve the accuracy of our approach by 1.8, see Tab. 3. Moreover, we show that the DDSC can be also applied to other semantic segmentation methods. The DDSC improve the accuracy of the RefineNet-101 [31] by 1.9 mean IoU. The RefineNet-101 with our DDSC achieves 45.5 mean IoU, which is almost 2% better than the RefineNet-101 with

Approach	**mean IoU**
FCN-32s HHA [36]	24.2
Gupta *et al.* [14] RGBD	27.4
Gupta *et al.* [15] RGBD	28.6
FCN-32s RGB [36]	29.2
FCN-32s RGBD [36]	30.5
FCN-32s RGB-HHA [36]	32.8
FCN-16s RGB-HHA [36]	34.0
Eigen [12]	34.1
Contextual Pairwise [32]	40.6
RefineNet-101 [31]	43.6
Ours (single scale)	**48.1**

Table 2: Comparison of our approach (using a single-scale evaluation) with the state-of-the-art (using a multi-scale evaluation) on the NYUD dataset.

Approach	**DDSC**	**Msc Eva**	**mIoU**
RefineNet-101 [31]			43.6
RefineNet-101 [31]		✓	44.7
* RefineNet-101	✓		45.5
RefineNet-152 [31]		✓	46.5
* Ours-101 (without CA)	✓		46.4
* Ours-101	✓		**48.1**
* Ours-101			46.3

Table 3: Experiments on the NYUD dataset. The * indicates our experiments. 101 and 152 indicate number of layers of an encoder. DDSC indicates our dense decoder shortcut connections. CA indicates our context aggregation. Msc Eva indicates multi-scale evaluation, as in [31].

a single-scale evaluation and almost 1% better than the RefineNet-101 with a time-consuming multi-scale evaluation. Furthermore, we show that our context aggregation step (convolution-pooling blocks) of the semantic feature generation stage contributes 1.7% to the accuracy. We also show that our ResNeXt-style modules increase accuracy by almost 2.6% *w.r.t.* the RefineNet-101. Finally, we report that the inference takes 0.3 second per image (640 $\times$ 480 pixels spatial resolution).

The **CamVid** [3] is a road scene understanding dataset with 11 semantic classes and 701 images (367 training, 101 validation and 233 testing). We resize images to 480 $\times$ 360 pixels for both training and testing, as in [2]. The results are presented in Tab. 4 and Fig. 5. Our approach using a single-scale evaluation achieves a new state-of-the-art result 70.9 mean IoU, outperforming by 2.9 the second best method, which is using multi-scale evaluation. Note that some of the methods in Tab. 4 use high resolution images, which expectedly would further improve our results.

Approach	**mean IoU**
DeconvNet [39]	48.9
SegNet [2]	50.2
FCN-8 [36]	52.0
DeepLab-LargeFOV-DenseCRF [5]	54.7
Bayesian SegNet [25]	63.1
Dilation [53]	65.3
Dilation + FSO [27]	66.1
FC-DenseNet103 [24], single scale	66.9
G-FRNet [23]	68.0
Ours (single scale)	**70.9**

Table 4: Comparison of our approach (using single-scale evaluation) with the state-of-the-art (most using multi-scale evaluation) on the CamVid dataset.

Approach	**mean IoU**
FCN [36]	62.2
Zoom-out [37]	69.6
DeepLab [5]	71.6
CRF-RNN [55]	72.0
DeconvNet [39]	72.5
G-CRF [4]	73.2
DPN [35]	74.1
Piecewise [32]	75.3
PSPNet [54]	**82.6**
Ours (single scale)	**81.2**

Table 5: Comparison of our approach (using single-scale evaluation) with the state-of-the-art (using multi-scale evaluation) on the Pascal VOC 2012 dataset. Methods are trained with Pascal VOC 2012 and SDB [16] data only.

The **Pascal VOC 2012** dataset [13] is the most popular semantic segmentation dataset, with 20 object categories and a background class. It contains 1464 training, 1449 validation and 1456 testing images. Following common practice [55, 5, 54], we use the training set which also includes additionally annotated training and validation VOC images [16]. As annotations of the test set are not publicly available, the final accuracy is obtained by submitting the predicted segmentations to the Pascal VOC challenge evaluation server [13]. Our results are presented in Tab. 5. Sample predictions of our approach are shown in Fig. 5. Our approach using a single-scale evaluation achieves the second best score against methods using multi-scale evaluations. Our algorithm is only outperformed by PSPNet [54] which, we believe, is due to multi-scale evaluation, but also due to the larger batch size used (1 image for us, 16 for PSPNet). Recent works [42, 11] showed that the batch normalization layer introduces additional non-linearity to the

Approach	mean IoU
CFM [10]	34.4
FCN-8s [36]	37.8
CRF-RNN [55]	39.3
ParseNet [34]	40.4
BoxSup [9]	40.5
Higher Order CRF [1]	41.3
Contextual Pairwise [32]	43.3
VeryDeep [50]	44.5
DeepLab [6]	45.7
Global Context Embedding [21]	46.5
RefineNet-101 [31]	47.1
RefineNet-152 [31]	47.3
Ours (single scale)	**47.8**

Table 6: Comparison of our approach (using single-scale evaluation) with the state-of-the-art (using multi-scale evaluation) on the Pascal-Context dataset.

Approach	mean IoU
Attention [7]	57.4
HAZN [51]	57.5
LG-LSTM [30]	58.0
Grap-LSTM [29]	60.2
DeepLab [5]	62.8
DeepLab-v2 (ResNet-101) [6]	64.9
Holistic [28], single scale	66.3
RefineNet-101 [31]	**68.6**
Ours (single scale)	**68.6**

Table 7: Comparison of our approach (using single-scale evaluation) with the state-of-the-art (most using multi-scale evaluation) on the Pascal Person-Part dataset.

network, so a larger batch size and a training strategy design taking advantage of this are likely to increase overall accuracy. Larger batch sizes are however not tractable in our single GPU scenario. Also connected to this point, the PSPNet encoder uses a modified ResNet backbone, which maintains feature maps of higher resolutions than the original ResNet (this may be another source of the improvement of the PSPNet, not tractable in our single-GPU scenario). Note that methods using additional data (*e.g.* IDW [49] and MS-COCO [33]) are excluded in this comparison to assure that all the methods are using the same data for training.

The **Pascal-Context** dataset [38] provides pixel-wise labels for the PASCAL VOC 2010 images. It contains 4998 training and 5105 testing images, and annotations for 60 categories (59 object categories and a background class). Our results are presented in Tab. 6. Sample predictions of our approach are shown in Fig. 5. Our approach using a single-scale evaluation achieves a new state-of-the-art result 47.8 mean IoU, outperforming many methods that are using multi-scale evaluation.

The **Pascal Person-Part** [8] is an object parsing dataset, which provides pixel-level labels for six person parts (head, torso, upper arms, lower arms, upper legs, and lower legs) and a background. There are 1717 training/validation and 1818 testing images. Our results are presented in Tab. 7. Sample predictions of our approach are shown in Fig. 4. Our approach using a single-scale evaluation achieves the state-of-the-art accuracy, the same as the RefineNet-101 using time-consuming multi-scale evaluation.

5. Conclusion

We have proposed a novel end-to-end trainable, deep, encoder-decoder architecture for single-pass semantic segmentation. Our approach is based on a cascaded architecture with feature-level long-range skip connections. We have incorporated the structure of ResNeXt's residual building blocks and adopted the strategy of repeating a building block that aggregates a set of transformations with the same topology. Moreover, we have proposed a novel decoder's architecture, consisting of blocks, each capturing context information, generating semantic features, and enabling fusion between different output resolutions. Crucially, we have introduced dense decoder shortcut connections to allow decoder blocks to use semantic feature maps from all previous decoder levels, *i.e.* from all higher-level feature maps. The dense decoder shortcut connections allow for effective information propagation from one decoder block to another, and for multi-level feature fusion that significantly improves the accuracy. We have performed an extensive evaluation of our approach on various semantic segmentation datasets, obtaining the state-of-the-art performance on several challenging datasets, without the need of time-consuming multi-scale averaging of previous works. Furthermore, our dense decoder shortcut connections have shown that they are likely to apply to other architectures as well.

Acknowledgement

Research presented in this paper has been supported by Dyson Technology Ltd.

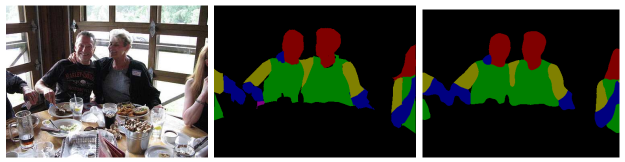

Figure 4: Person-Part: sample input image (left), ground truth (middle) and prediction of our approach (right).

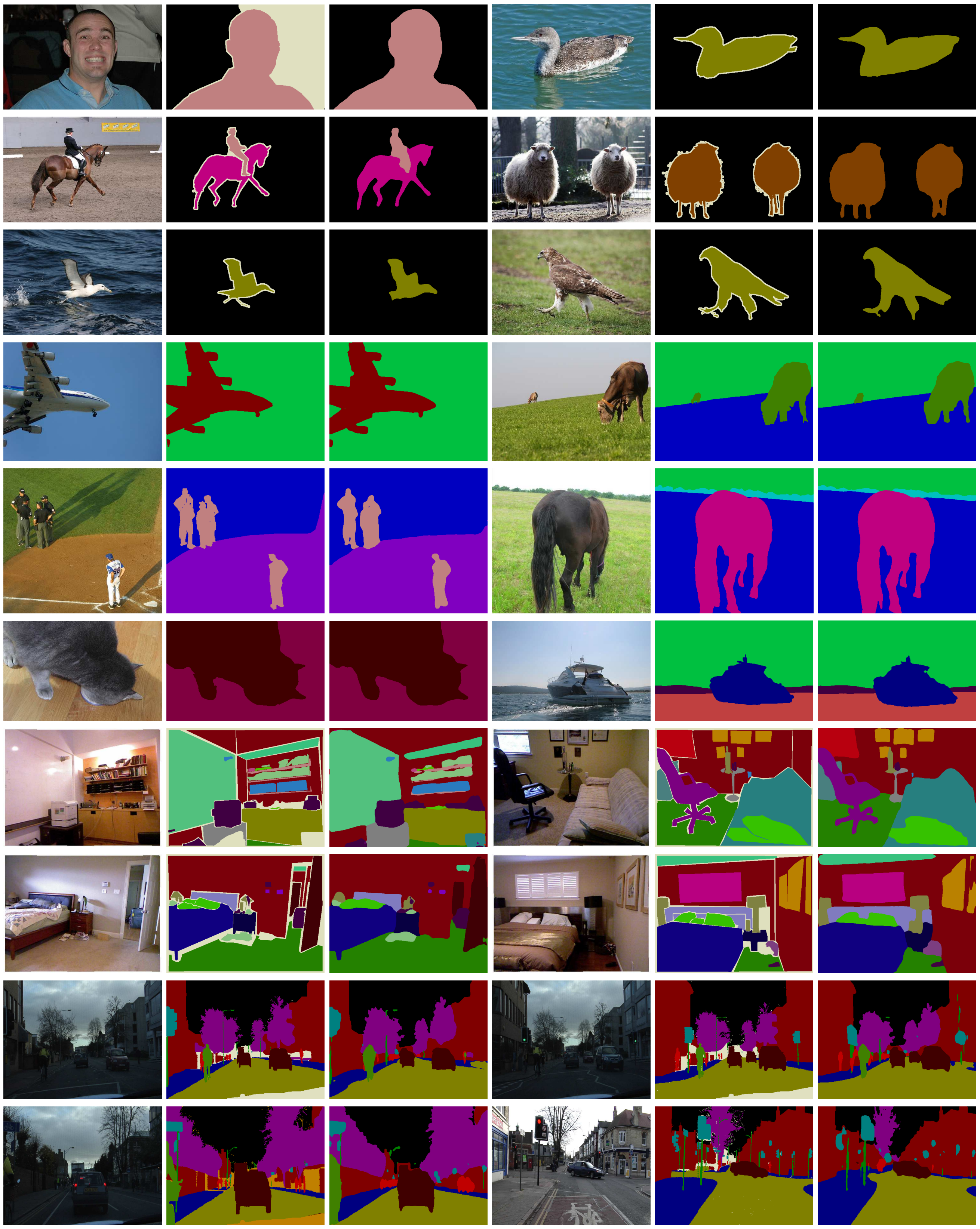

Figure 5: Sample predictions of our approach on Pascal VOC 2012 (1st-3rd rows), Pascal-Context (4th-6th rows), NYUD (7th and 8th rows) and CamVid datasets (9th and 10th rows). From left to right: input image, ground truth and our prediction.

References

[1] A. Arnab, S. Jayasumana, S. Zheng, and P. H. Torr. Higher Order Conditional Random Fields in Deep Neural Networks. In *ECCV*, 2016.

[2] V. Badrinarayanan, A. Kendall, and R. Cipolla. SegNet: A Deep Convolutional Encoder-Decoder Architecture for Image Segmentation. *TPAMI*, 39(12):2481–2495, 2017.

[3] G. Brostow, J. Shotton, J. Fauqueur, and R. Cipolla. Segmentation and Recognition using Structure from Motion Point Clouds. In *ECCV*, 2008.

[4] S. Chandra and I. Kokkinos. Fast, Exact and Multi-Scale Inference for Semantic Image Segmentation with Deep Gaussian CRFs. In *ECCV*, 2016.

[5] L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille. Semantic Image Segmentation with Deep Convolutional Nets and Fully Connected CRFs. In *ICLR*, 2015.

[6] L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille. DeepLab: Semantic Image Segmentation with Deep Convolutional Nets, Atrous Convolution, and Fully Connected CRFs. *arXiv preprint arXiv:1606.00915*, 2016.

[7] L.-C. Chen, Y. Yang, J. Wang, W. Xu, and A. L. Yuille. Attention to Scale: Scale-aware Semantic Image Segmentation. In *CVPR*, 2016.

[8] X. Chen, R. Mottaghi, X. Liu, S. Fidler, R. Urtasun, and A. Yuille. Detect What You Can: Detecting and Representing Objects using Holistic Models and Body Parts. In *CVPR*, 2014.

[9] J. Dai, K. He, and J. Sun. BoxSup: Exploiting Bounding Boxes to Supervise Convolutional Networks for Semantic Segmentation. In *ICCV*, 2015.

[10] J. Dai, K. He, and J. Sun. Convolutional Feature Masking for Joint Object and Stuff Segmentation. In *CVPR*, 2015.

[11] V. Dumoulin, J. Shlens, and M. Kudlur. A Learned Representation For Artistic Style. In *ICLR*, 2017.

[12] D. Eigen and R. Fergus. Predicting Depth, Surface Normals and Semantic Labels with a Common Multi-Scale Convolutional Architecture. In *ICCV*, 2015.

[13] M. Everingham, L. Van Gool, C. K. Williams, J. Winn, and A. Zisserman. The PASCAL Visual Object Classes (VOC) Challenge. *IJCV*, 88(2):303–338, 2010.

[14] S. Gupta, P. Arbelaez, and J. Malik. Perceptual Organization and Recognition of Indoor Scenes from RGB-D Images. In *CVPR*, 2013.

[15] S. Gupta, R. Girshick, P. Arbelaez, and J. Malik. Learning Rich Features from RGB-D Images for Object Detection and Segmentation. In *ECCV*, 2014.

[16] B. Hariharan, P. Arbelaez, R. Girshick, and J. Malik. Simultaneous Detection and Segmentation. In *ECCV*, 2014.

[17] B. Hariharan, P. Arbelaez, R. Girshick, and J. Malik. Hypercolumns for Object Segmentation and Fine-grained Localization. In *CVPR*, 2015.

[18] K. He, X. Zhang, S. Ren, and J. Sun. Deep Residual Learning for Image Recognition. In *CVPR*, 2016.

[19] K. He, X. Zhang, S. Ren, and J. Sun. Identity Mappings in Deep Residual Networks. In *ECCV*, 2016.

[20] G. Huang, Z. Liu, L. van der Maaten, and K. Q. Weinberger. Densely Connected Convolutional Networks. In *CVPR*, 2017.

[21] W.-C. Hung, Y.-H. Tsai, X. Shen, Z. Lin, K. Sunkavalli, X. Lu, and M.-H. Yang. Scene Parsing With Global Context Embedding. In *ICCV*, 2017.

[22] S. Ioffe and C. Szegedy. Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift. In *ICML*, 2015.

[23] M. A. Islam, M. Rochan, N. D. B. Bruce, and Y. Wang. Gated Feedback Refinement Network for Dense Image Labeling. In *CVPR*, 2017.

[24] S. Jegou, M. Drozdzal, D. Vazquez, A. Romero, and Y. Bengio. The One Hundred Layers Tiramisu: Fully Convolutional DenseNets for Semantic Segmentation. In *CVPR Workshop*, 2017.

[25] A. Kendall, V. Badrinarayanan, and R. Cipolla. Bayesian SegNet: Model Uncertainty in Deep Convolutional Encoder-Decoder Architectures for Scene Understanding. In *BMVC*, 2017.

[26] I. Kokkinos. Pushing the Boundaries of Boundary Detection using Deep Learning. In *ICLR*, 2016.

[27] A. Kundu, V. Vineet, and V. Koltun. Feature Space Optimization for Semantic Video Segmentation. In *CVPR*, 2016.

[28] Q. Li, A. Arnab, and P. H. Torr. Holistic, Instance-level Human Parsing. In *BMVC*, 2017.

[29] X. Liang, X. Shen, J. Feng, L. Lin, and S. Yan. Semantic Object Parsing with Graph LSTM. In *ECCV*, 2016.

[30] X. Liang, X. Shen, D. Xiang, J. Feng, L. Lin, and S. Yan. Semantic Object Parsing with Local-Global Long Short-Term Memory. In *CVPR*, 2016.

[31] G. Lin, A. Milan, C. Shen, and I. Reid. RefineNet: Multi-Path Refinement Networks for High-Resolution Semantic Segmentation. In *CVPR*, 2017.

[32] G. Lin, C. Shen, A. van den Hengel, and I. Reid. Efficient Piecewise Training of Deep Structured Models for Semantic Segmentation. In *CVPR*, 2016.

[33] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollar, and C. L. Zitnick. Microsoft COCO: Common Objects in Context. In *ECCV*, 2014.

[34] W. Liu, A. Rabinovich, and A. C. Berg. ParseNet: Looking Wider to See Better. In *ICLR*, 2016.

[35] Z. Liu, X. Li, P. Luo, C.-C. Loy, and X. Tang. Semantic Image Segmentation via Deep Parsing Network. In *ICCV*, 2015.

[36] J. Long, E. Shelhamer, and T. Darrell. Fully Convolutional Networks for Semantic Segmentation. In *CVPR*, 2015.

[37] M. Mostajabi, P. Yadollahpour, and G. Shakhnarovich. Feedforward semantic segmentation with zoom-out features. In *CVPR*, 2015.

[38] R. Mottaghi, X. Chen, X. Liu, N.-G. Cho, S.-W. Lee, S. Fidler, R. Urtasun, and A. Yuille. The Role of Context for Object Detection and Semantic Segmentation in the Wild. In *CVPR*, 2014.

[39] H. Noh, S. Hong, and B. Han. Learning Deconvolution Network for Semantic Segmentation. In *ICCV*, 2015.

[40] P. O. Pinheiro, R. Collobert, and P. Dollar. Learning to Segment Object Candidates. In *NIPS*, 2015.

[41] P. O. Pinheiro, T.-Y. Lin, R. Collobert, and P. Dollar. Learning to Refine Object Segments. In *ECCV*, 2016.

[42] S.-A. Rebuffi, H. Bilen, and A. Vedaldi. Learning multiple visual domains with residual adapters. In *NIPS*, 2017.

[43] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and L. Fei-Fei. ImageNet Large Scale Visual Recognition Challenge. *IJCV*, 115(3):211–252, 2015.

[44] J. Shotton, M. Johnson, and R. Cipolla. Semantic Texton Forests for Image Categorization and Segmentation. In *CVPR*, 2008.

[45] J. Shotton, T. Sharp, A. Kipman, A. Fitzgibbon, M. Finocchio, A. Blake, M. Cook, and R. Moore. Real-Time Human Pose Recognition in Parts from Single Depth Images. *CACM*, 56(1):116–124, 2013.

[46] J. Shotton, J. Winn, C. Rother, and A. Criminisi. Texton-Boost for Image Understanding: Multi-Class Object Recognition and Segmentation by Jointly Modeling Texture, Layout, and Context. *IJCV*, 81(1):2–23, 2009.

[47] N. Silberman, D. Hoiem, P. Kohli, and R. Fergus. Indoor Segmentation and Support Inference from RGBD Images. In *ECCV*, 2012.

[48] K. Simonyan and A. Zisserman. Very Deep Convolutional Networks for Large-Scale Image Recognition. In *ICLR*, 2015.

[49] G. Wang, P. Luo, L. Lin, and X. Wang. Learning Object Interactions and Descriptions for Semantic Image Segmentation. In *CVPR*, 2017.

[50] Z. Wu, C. Shen, and A. van den Hengel. Bridging Category-level and Instance-level Semantic Image Segmentation. *arXiv preprint arXiv:1605.06885*, 2016.

[51] F. Xia, P. Wang, L.-C. Chen, and A. L. Yuille. Zoom Better to See Clearer: Human and Object Parsing with Hierarchical Auto-Zoom Net. In *ECCV*, 2016.

[52] S. Xie, R. Girshick, P. Dollar, Z. Tu, and K. He. Aggregated Residual Transformations for Deep Neural Networks. In *CVPR*, 2017.

[53] F. Yu and V. Koltun. Multi-Scale Context Aggregation by Dilated Convolutions. In *ICLR*, 2015.

[54] H. Zhao, J. Shi, X. Qi, X. Wang, and J. Jia. Pyramid Scene Parsing Network. In *CVPR*, 2017.

[55] S. Zheng, S. Jayasumana, B. Romera-Paredes, V. Vineet, Z. Su, D. Du, C. Huang, and P. H. Torr. Conditional Random Fields as Recurrent Neural Networks. In *ICCV*, 2015.