Laplacian Reconstruction and Refinement for Semantic Segmentation

Golnaz Ghiasi and Charless C. Fowlkes
{gghiasi,fowlkes}@ics.uci.edu
Dept. of Computer Science,
University of California, Irvine

Abstract. CNN architectures have terrific recognition performance but rely on spatial pooling which makes it difficult to adapt them to tasks that require dense pixel-accurate labeling. This paper makes two contributions: (1) We demonstrate that while the apparent spatial resolution of convolutional feature maps is low, the high-dimensional feature representation contains significant sub-pixel localization information. (2) We describe a multi-resolution reconstruction architecture, akin to a Laplacian pyramid, that uses skip connections from higher resolution feature maps to successively refine segment boundaries reconstructed from lower resolution maps. This approach yields state-of-the-art semantic segmentation results on PASCAL without resorting to more complex CRF or detection driven architectures.

Keywords: Semantic Segmentation, Convolutional Neural Networks

1 Introduction

Deep convolutional neural networks (CNNs) have been highly effective at semantic segmentation due to the capacity of pre-trained feature hierarchies to model and recognize objects and “stuff”. This has allowed them to significantly outperform previous approaches (e.g., [123]) that relied on hand-designed features and recognizers trained from scratch. A key difficulty in the straightforward adaption of CNN features to segmentation is that spatial pooling architectures result in representations with reduced spatial resolution. This tradeoff can be viewed as a direct result of introducing invariance to spatial deformations which is needed for strong recognition performance. From a signal processing perspective, classifiers built on activations at high levels of the convolutional feature hierarchy have large receptive fields in the input domain whose linearized response on natural image data is dominated by low frequencies, yielding a smooth output that can be sub-sampled with little loss in classification accuracy.

In this paper, we investigate this spatial-semantic uncertainty principle for CNN hierarchies (see Fig.1). We begin with the question of how much spatial information is represented at high levels of the feature hierarchy. A simple answer is that a given spatial location in a convolutional feature map corresponds to a large block of input pixels and an even larger “receptive field”. However, the
Fig. 1. In this paper, we explore the trade-off between spatial and semantic accuracy within CNN feature hierarchies. We and others have observed a spatial-semantic uncertainty principle in which high levels of the hierarchy make accurate semantic predictions but are poorly localized in space while at low levels, boundaries are precise but labels are noisy. We develop reconstruction techniques for increasing spatial accuracy at a given level and refinement techniques for fusing multiple levels that limit these tradeoffs and produce improved semantic segmentations.

vector of activations at a given location is sparse and high-dimensional which allows, in principle, the possibility for encoding substantial amounts of “sub-pixel” spatial information. In particular, while max pooling a single channel clearly destroys spatial information in that channel, spatial filtering introduces strong correlations across channels which could preserve spatial information implicitly. We show that this indeed the case and demonstrate a simple approach to spatial decoding that substantially improves over commonly used bilinear upsampling schemes (see Fig. 2).

Having squeezed more spatial information out of a given layer of the hierarchy, we turn to the question of fusing information across layers. A standard approach has been to either concatenate features (e.g., [4]) or linearly combine predictions (e.g., [5]). Concatenation is appealing but suffers from the high dimensionality of the resulting features. On the other hand, simply combining predictions from multiple layers does not make good use of the relative spatial-semantic content tradeoff. High-resolution layers are shallow with small receptive fields and hence yield inherently noisy predictions with high pixel-wise loss. As a result, they are significantly down-weighted relative to low-resolution layers during linear fusion and have little effect on final predictions.

Inspired by recent work on residual networks [6], we propose an architecture in which predictions derived from high-resolution layers are only required to correct residual errors in the low-resolution prediction. Importantly, we use
Fig. 2. (a) Bilinear upsampling architecture for FCN32s network (left) and our reconstruction network (right). (b) Example of Class conditional probability maps and semantic segmentation predictions from FCN32s which performs bilinear upsampling (middle) and our 32x reconstruction network (right).

multiplicative gating to avoid integrating (and hence penalizing) high-resolution outputs in regions where the low-resolution predictions are confident about the semantic content. We call our method Laplacian Reconstruction and Refinement (LRR) since the final architecture closely resembles the Laplacian reconstruction pyramid [7]. Indeed, the class scores predicted at each level of our architecture typically look like bandpass decomposition of the full resolution segmentation mask (see Fig. 3).

2 Related Work

This inherent lack of spatial detail in CNN feature maps has been attacked using a variety of techniques. One approach is to combine CNN predictions with a conditional random field (CRF) that utilizes bottom-up superpixels [8,9] or generic boundary detection [10,11] to precisely localize segment boundaries. These pairwise features have been successfully integrated with CNN unary class predictions [12,13,14], typically using a fully connected CRF [15] or closely related architectures such as domain transfer [11]. Training such a CRF can be carried out using end-to-end back-propagation [16].

Our results are related to an alternate line of attack that aims to directly improve the spatial resolution of CNN activation maps. One insight is that spatial information lost during max-pooling can in part be recovered by unpooling and deconvolution [17] providing a useful way to visualize input dependency in feed-forward models [18]. This idea has been developed using learned deconvolution filters to perform semantic segmentation [19]. Similar to our approach, the deconvolution network uses learned deconvolution filters that map from higher to lower dimensional activations. However, the deeply stacked deconvolutional output layers are difficult to train, requiring multi-stage training and more complicated object proposal aggregation.
4 G. Ghiasi and C. Fowlkes

Fig. 3. Overview of our Laplacian reconstruction network architecture. We use low-resolution feature maps in the CNN hierarchy to reconstruct a coarse, low-frequency segmentation map and then refines this map by adding in higher frequency details derived from higher-resolution feature maps. Boundary masking (inset) suppresses the contribution of higher resolution layers in areas where the segmentation is confident, allowing the reconstruction to focus on predicting the residual in uncertain areas (e.g., precisely localizing object boundaries).

A second key insight is that while activation maps at lower-levels of the CNN hierarchy lack object specificity, they do contain higher spatial resolution information. Performing classification using a “jet” of feature map responses aggregated across multiple layers has been successfully leveraged for semantic segmentation [5], generic boundary detection [20], simultaneous detection and segmentation [4], and scene recognition [21]. Our architecture shares the basic skip connections of [5]. However, we change substantially how the layers are fused into a final segmentation.

We note that our results are complementary to work on models that incorporate object proposals [22][19], attentional and scale selection mechanisms [23] and CRF-style spatial filtering. In many cases, our approach is simply a drop in replacement and any of these methods can benefit from the improved spatial prediction offered by our proposed architecture.

3 Reconstruction with learned basis functions

Bilinear upsampling is a standard approach for computing high resolution segmentation maps from a low resolution feature map [4][12][16][23][22][24]. Typically a convolutional layer is used to first compute a low resolution class score from the feature map. Then this low resolution class score is upsampled to a high resolution score map using bilinear filter. (see Fig. 2(a)).
Fig. 4. Efficient pyramidal implementation of our Laplacian reconstruction network. At each resolution layer, the reconstruction filters perform the same amount of upsampling (e.g., 8x). An additional 2x bilinear upsampling is then applied to each class score map before combining it with higher resolution predictions. Compared to the architecture in Fig. 3, this pyramidal implementation has fewer parameters (smaller reconstruction filters) and is more efficient in terms of memory and computation with little loss in segmentation accuracy.

We believe that first computing class scores at low-resolution and then upsampling limits the use of localization information that might be coded in across the many channels of the low resolution feature map. Upsampling the whole feature map prior to classification poses difficulties in terms of computational constraints due to the large number of feature channels (e.g. 4096). It is also worth pointing out that bilinear upsampling commutes with 1x1 convolutions used for class prediction so performing per-pixel linear classification on an upsampled feature map would yield equivalent results unless additional rounds of filtering and rectification were carried out on the high-resolution feature map.

In order to retain more detailed spatial information, we avoid collapsing the high-dimensional feature map down to low-resolution class predictions. Instead, we encode the spatial pattern of high-resolution class scores using a linear combination of high-resolution basis functions whose weights are predicted from the high-dimensional feature map (see Fig. 2 (a)). We term this approach “reconstruction” to distinguish it from the standard bilinear upsampling (although upsampling can clearly be seen as special case with a single tent-shaped basis function).
Fig. 5. Category specific basis functions for reconstruction. Note that the basis functions for different category are adapted to modeling that class. eg. basis functions for airplane are adapted to modeling airplane segments which tend to be elongated in the horizontal direction while the ones for bottle elongated in the vertical direction.

In our implementation, we divide the high resolution score maps into overlapping blocks whose size is set based on number of max-pooling layers. (e.g., for 5 rounds of 2x2 max pooling the size of non-overlapping blocks would be 32x32 while our overlapping blocks are 64x64). We use a convolution layer to predict the basis coefficients for reconstructing each of these blocks from the high-dimensional, low-resolution feature map. These coefficients of each block and each class are multiplied by the set of basis function for the class and summed using a standard deconvolution layer to generate the desired full resolution class scores.

3.1 Connection to spline interpolation

We note that an alternative approach would be to perform higher-order spline interpolation during upsampling. For example, the basis functions could be chosen to be a standard set of non-overlapping polynomial basis functions. Given the low resolution class score map, the coefficients for each basis function could be computed by convolving the score map with a set of filters. The support of these filters would be tied to the spline order (e.g., 1x1 for bilinear, 3x3 for bicubic) while the filter weights could be set analytically to assure continuity between neighboring reconstructions. While we use learned filters and basis functions, this connection informs our architecture. In particular, our choice of 5x5x4096 filter kernels for predicting the coefficient associated with each basis stems from the notion of introducing linear dependencies between neighboring basis weights in order to improve continuity of the output predictions.
3.2 Learning basis functions

To leverage limited amounts of training data and speed up training, we initialized the deconvolution layers with a meaningful set of filters for reconstruction. Since our goal is accurate reconstruction of segmentation maps with a low-dimensional basis, PCA provides a natural tool. For this purpose, we extract 10000 patches per class of size $32 \times 32$ from the PASCAL VOC 2011 training data. For each class, we only extract patches for which at least 2% of the patch belongs to the class. We apply PCA on the extracted patches to compute a class specific set of initial basis filters. Example of basis functions for different categories are shown in Fig. 5. Interestingly, there is some significant variation among classes due to different segment shape statistics.

Our LRR architecture requires basis functions for each resolution feature map. However, due to the self-similar pyramid construction (described below), the statistics of class predictions at each layer are quite similar. We thus found it sufficient to initialize the basis functions at higher-resolution layers by simply sub-sampling the basis functions learned for the coarse layer.

We experimented with varying the number of basis functions but found that 10 were sufficient for accurate reconstruction of the class scores. Models trained with more than 10 basis functions commonly predicted zero weight coefficients for the higher-frequency basis functions. This suggests that we may be hitting the limit of how much spatial information can be extracted from the low-res feature map (i.e., roughly 3x more than bilinear). However, this estimate is only a lower-bound since there are obvious limitations to how well we can fit the model and other generative architectures (e.g., using larger dictionaries sparse basis functions) or additional information (e.g., max pooling “switches” in deconvolution\[17\]) may well do better.

4 Laplacian Refinement

The basic intuition for our multi-level architecture comes from Burt and Adelson’s Laplacian Pyramid [7], which decomposes an image into disjoint frequency bands using an elegant recursive computation (analysis) that produces appropriately down-sampled sub-bands such that the sum of the resulting sub-bands (synthesis) perfectly reproduces the original image. While the notion of frequency response is not appropriate for the non-linear filtering performed by standard CNNs, casual inspection of the response of individual activations to shifted input images reveals a power spectral density whose high-frequency components decay with depth leaving primarily low-frequency components (with a few high-frequency artifacts due to disjoint bins used in pooling). This suggests a loose analogy in which the basic CNN architecture serves the role of the analysis pyramid while the segmentation estimation acts in reverse as the synthesis pyramid.

Figure 3 shows the overall architecture of our model which attempts to make this analogy concrete. Starting from the coarse scale “low-frequency” segmentation estimate, we carry out a sequence of successive refinements, adding in
Fig. 6. Visualization of segmentation results of our model with and without boundary masking. For each row, we show the input image, ground-truth and the segmentation results of 32x and 8x layers of our model without masking (middle) and with masking (right). The segmentation results for 8x layer of the model without masking has some noise while its 32x outputs don’t include them. Masking allows such noise to be repressed in regions where the 32x outputs have high confidence.

information from “higher-frequency” sub-bands to improve the spatial fidelity of the resulting segmentation masks. For example, since the 32x layer already captures the coarse scale support of the object, prediction from the 16x layer does not need to include this information and can instead focus on adding finer scale refinements of the segment boundary.

4.1 Boundary masking

In practice, simply adding together the outputs of the analysis layers does not yield the desired effect. Unlike the Laplacian image analysis pyramid, the high resolution feature maps of the CNN do not have the “low-frequency” content subtracted out – in fact, a better analogy for the CNN is a Gaussian pyramid! As Fig.1 shows, high-resolution layers still happily make “low-frequency” predictions (e.g., in the middle of a large segment) even though they are often incorrect. As a result, in an architecture that simply sums together predictions across layers, we found the learned parameters tend to down-weight the contribution of high-resolution predictions to the sum in order to limit the potentially disastrous effect of these noisy predictions. However, this also hampering the ability of the high-resolution predictions to refine the segmentation in areas containing high-frequency content (i.e., segment boundaries).

In order to remedy this, we introduce a masking step which serves to explicitly “subtract out” the “low-frequency” content from the high-resolution signal. In practice this takes the form of a multiplicative gating that prevents the high-resolution predictions from contributing to the final response in regions where lower-resolution predictions are confident. The inset in Fig.3 shows how this
boundary mask is computed by using max pooling to dilate the confident foreground and background predictions and taking their difference to isolate the boundary. The size of this dilation is tied to the resolution reconstruction (i.e., decreasing by 2x at each successive layer). A closely related architecture was used in [25] for generative synthesis where the output of a lower-resolution model was used as input for a CNN which predicted an additive refinement.

4.2 Pyramid architecture

In the architecture introduced in the last section (Fig. 3), coarse-scale reconstructions are represented at the full image resolution. For purposes of comparing different architectures as described in the experiments, we found it useful to represent segmentation class scores at the full image resolution. However, since the predictions from the coarse scale levels are necessarily quite smooth, it is quite natural to represent them at lower-resolution and only up-sample them as necessary. Fig. 4 shows such a pyramidal variant of our architecture which utilizes a lower-resolution set of reconstruction basis filters and then perform an additional bilinear upsampling step immediately prior to summing in the class score from the next higher resolution layer.

This implementation has several advantages in terms of memory and computational requirements. This is especially true during training since the loss functions for low-resolution layers are computed over class probability maps and corresponding down-sampled ground-truth annotations. The total number of model parameters is also decreased in the pyramidal architecture. Since the reconstruction filters perform the same amount of upsampling at every layer, the size of the reconstruction basis filters is kept small (e.g., 16x16 for Fig. 4) relative to the non-pyramid architecture where the filters are as large as 64x64 for the 32x feature map. We found that this pyramidal architecture yielded nearly the same performance while using significantly less memory.

4.3 Stage-wise training

Since our proxy for frequency content of a given layer is based on the class probability predictions at that layer, it is quite natural to train the model in a stage-wise fashion. We use a pixel-wise softmax log loss defined at the original image resolution, optimize the 32x resolution class predictions, then add in connections to the 16x layer, and continue to fine tune. As we add in additional layers, we retain the loss functions for the coarser layers. This can be seen as a form of deep supervision [20,26] that serves to anchor the function of previously trained layers.

5 Experiments

5.1 Dataset

We use PASCAL VOC segmentation dataset for training and testing our model and the baselines. This dataset consists of 20 foreground object classes and
| Model                  | Pixel Acc. | Mean Acc. | Mean IU  |
|------------------------|------------|-----------|----------|
| FCN-32s                | 89.1%      | 73.3%     | 59.4%    |
| FCN-16s                | 90.0%      | 75.7%     | 62.4%    |
| FCN-8s                 | 90.3%      | 75.9%     | 62.7%    |
| LRR-32x (w/o aug)      | 90.7%      | 78.9%     | 64.1%    |
| LRR-32x                | 91.5%      | 81.6%     | 66.8%    |
| LRR-16x                | 91.8%      | 81.6%     | 67.8%    |
| LRR-8x                 | 92.4%      | 83.2%     | 69.5%    |
| LRR-4x                 | 92.2%      | 83.7%     | 69.0%    |
| LRR-4x-ms              | 92.8%      | 84.6%     | 71.4%    |

**Fig. 7.** Comparison of our segment reconstruction model, LRR (without boundary masking) and the baseline FCN model [5] which uses bilinear upsampling. We find consistent benefits from using a higher-dimensional reconstruction basis rather than upsampling class prediction maps. We also see improved performance from using multi-scale training augmentation, fusing multiple feature maps, and running on multiple scales at test time. Note that the performance benefit of fusing multiple resolution feature maps diminishes with no gain or even decrease performance from adding in the 4x layer. Boundary masking (cf. Fig. 8) allows for much better utilization of these fine scale features.

one background class. Our models are trained on training/validation set split specified by [27]. This includes 11287 training images. For validation, we use the subset of 736 images from PASCAL VOC 2011 val examples that are not included in training. We conduct our experiments on the model architecture using this validation data and test our final model via submission to the PASCAL VOC 2012 test data server which benchmarks on a set of 1456 images. We focus primarily on the average Intersection-over-Union metric which generally provides a more sensitive performance measure than per-pixel or per-class accuracy.

### 5.2 Parameter Optimization

We augment the layers of the pre-trained VGG-16 network [28] with our LRR architecture and fine-tune all layers via back-propagation. All models were trained and tested with Matconvnet [29] on a single NVIDIA GPU. The models and code will be released on publication.

We use standard stochastic gradient descent with batch size of 20, momentum of 0.9 and weight decay of 0.0005. We use learning rates of 0.005 to train 32x network. When we add 16x, 8x and 4x branches, we change the learning rate to 0.005/2, 0.005/3 and 0.005/4, respectively to account for the increase in total loss produced by adding additional terms to the objective.

We initialize filters in the convolution layers that predict reconstruction coefficients to 0. We set the deconvolution filters equal to the learned bases using PCA for each class. In our initial experiments we found that these basis functions did not change significantly from their initialization during learning and ultimately set the learning rate for these filters to zero to speed up training.
Laplacian reconstruction and refinement

| VOC 2011-val | training | VOC+COCO |
|--------------|----------|----------|
|              | unmasked | masked   |
| LRR-4x(32x) | 67.4%    | 67.5%    | 70.4%    | 70.4%    |
| LRR-4x(16x) | 68.6%    | 69.0%    | 71.4%    | 71.9%    |
| LRR-4x(8x)  | 69.1%    | 70.1%    | 72.0%    | 72.8%    |
| LRR-4x      |          |          |          |          |
| LRR-4x-ms   | 69.0%    | 70.2%    | 72.0%    | 72.9%    |
| LRR-4x-ms-crf| 71.4%    | 72.1%    | 73.8%    | 74.4%    |
| LRR-4x      |          |          |          |          |

![Table](image)

**Fig. 8.** Mean intersection-over-union (IOU) accuracy for intermediate outputs at different levels of our Laplacian reconstruction architecture trained with and without boundary masking. Masking allows us to squeeze additional gains out of high-resolution feature maps by focusing only on low-confidence areas near segment boundaries. Running the model at multiple scales and performing post-processing using a CRF yielded further performance improvements.

### 5.3 Multi-scale Data Augmentation

We augmented the training data with multiple scaled versions of each training examples. We randomly select an image size between 288 to 576 for each batch and then scale training examples of that batch to the selected size. When the selected size is larger than 384, we crop a window with size of 384×384 from the scaled image. This augmentation is helpful in improving the accuracy of the model and increased mean IOU of our 32x model from 64.07% to 66.81% on the validation data (see Fig.7).

### 5.4 Reconstruction vs Upsampling

To isolate the effectiveness of our proposed reconstruction method relative to simple upsampling, we compare the performance of our model without masking to the fully convolutional net (FCN) of [5]. For this experiment, we trained our model without scale augmentation using exactly same training data used for training the FCN models. We observed significant improvement over bilinear upsampling using reconstruction with 10 basis filters. Our 32x reconstruction model achieved a mean IOU of 66.81% while FCN-32s and FCN-8s had a mean IOU of 59.4% and 62.7%, respectively (Fig.7).

### 5.5 Boundary Masking

Lower resolution feature maps is computed using more number of max-pooling and as a result when they are used to compute semantic segmentation the output is coarser. But, at the same time it is more robust and selected class for a region is more accurate. We use the output from lower resolution feature maps to mask the predictions computed from higher resolution feature maps. This masking removes the wrong class predictions from higher resolution feature maps in the
regions that are not close to the boundaries. Fig. [6] demonstrates the effect of boundary masking. While the 32x prediction for both models do not include noises, the 4x prediction for the model without masking include some small regions with wrong class predictions. We compute mean IOU benchmarks for different outputs of our 4x model with and without masking (Table [8]). Boundary masking achieves a 1% overall improvement relative to a model without masking.

5.6 Evaluation near Object Boundaries

Our proposed model uses the higher resolution feature maps to refine the segmentation in the regions close to the boundaries of the objects and this results in a more detailed segmentation (see Fig. [11]). However, boundaries constitute a relatively small fraction of the image pixels, limiting the effect of these improvements on the overall IOU performance benchmark (see, e.g. Fig. [8]). To better characterize performance differences between models, we also compute mean IOU restricted to a narrow band of pixels around the ground-truth boundaries. This partitioning into figure/boundary/background is sometimes referred to as a tri-map in the matting literature and has been previously utilized in analyzing semantic segmentation performance [12,30].

Fig. [9] shows the mean IOU as a function of the width of the boundary zone for our LRR-4x model. We plot both the absolute performance and performance relative to the low-resolution 32x output. As the curves confirm, adding in higher resolution feature maps results in the most performance gain near object boundaries. Masking has the most noticeable effect far from object boundaries where it prevents high-resolution feature maps from corrupting accurate low-resolution predictions.
Fig. 10. Per-class mean intersection-over-union (IOU) performance on PASCAL VOC 2012 segmentation challenge test data. We evaluate models trained using only VOC training data as well as those trained with additional training data from COCO. Our model outperforms many existing approaches with the exception of Deep Parsing Network \[14\] and the Piecewise structured model of \[13\] which involve considerably more complicated architectures (e.g., based on deep CRFs).

5.7 CRF Post-processing

To show our architecture can easily be integrated with CRF-based models, we evaluated the the average LRR model predictions as a unary potential in a fully-connected CRF \[15][11\]. We resize each input image to three different scales (1.0,8.0,6.), apply the LRR model and then compute the pixel-wise maximum of predicted class conditional probability maps. Post-processing with the CRF yields small additional gains in performance. Fig. 8 reports the mean IOU for our basic LRR-4x model prediction when running at multiple scales and with the integration of the CRF. Fusing multiple scales yields a noticeable improvement (2\%) while the CRF gives an additional 1\% gain. We note that boundary masking and the CRF are somewhat redundant. The benefit of the CRF the unmasked model (1.56\%) is more than the one that utilizes masking (1.06\%). We believe this is because masking alone is able to remove some of the same noisy, spatially incoherent predictions targeted by the CRF (see qualitative examples in Fig. 6).

5.8 Additional Training Data

As the table \[10\] indicates, the current top performing architectures all use additional training data from the Microsoft COCO dataset \[31\]. To compare our model with these architectures, we also pre-trained our model on Microsoft COCO dataset. We utilized the 20 categories in COCO that are also present in PASCAL VOC and treat annotated objects from other categories as background. We selected images from the Microsoft COCO training and validation
set where at least 0.02% of the ground truth segmentation pixels are marked with classes labels present in the PASCAL VOC dataset. With this selection, we ended up using 97765 out of 123287 images of COCO training and validation set.

Training was performed in two stages. In the first stage, we trained LRR-32x on VOC images and Microsoft COCO images together. Since, COCO segmentation annotations are often coarser in comparison to VOC segmentation annotations, we did not use COCO images for training the LRR-16x, LRR-8x and LRR-4x. In the second stage, we used only PASCAL VOC images to further fine-tune the LRR-32x and then added in connections to the 16x, 8x and 4x layers and continue to fine-tune. We used the multi-scale data augmentation described in section 5.3 for both stages. Training on this additional data improved the mean IOU of our model from 73.1% to 75.2% on PASCAL VOC 2011 validation set (see Table 8).

5.9 Comparison to state-of-the-art

We compare our model performance with the state-of-the-art methods on the PASCAL VOC 2012 test set. Table 10 reports the mean IOU for our model when trained on only PASCAL VOC training data and when trained using both COCO and PASCAL VOC. In both cases our model outperforms existing approaches with the exception of Deep Parsing Network [14] and the Piecewise structured model of [13]. Our model achieves mean IOU of 76.8% which is within of 1% from the top performing model.

6 Discussion and Conclusions

We have presented a simple semantic segmentation system that utilizes two simple, extensible ideas (1) sub-pixel upsampling using a class-specific reconstruction basis, (2) a multi-level Laplacian pyramid reconstruction architecture that uses gating to more efficiently blend semantic-rich low-resolution feature map predictions with spatial detail from high-resolution feature maps. The resulting model is simple to train and achieves performance on VOC 2012 test which beats all but two recent models based on deep CRFs.

We note that both these CRF based methods, which tackle the difficult problem of building a CRF with deep pairwise potentials, involve considerably more elaborate architectures that unroll mean-field inference into a feed-forward network [14] or perform multiple rounds of CRF-based refinement [13], utilize richer features and require somewhat elaborate stage-wise incremental training to achieve good performance. We expect the relative simplicity and extensibility of our approach (along with its superior performance in cow, airplane, motorbike and train segmentation) will make it a ready candidate for further development or direct integration into more elaborate inference models.

Acknowledgements: This work was supported by NSF grant IIS-1253538.
Fig. 11. Examples of semantic segmentation results on PASCAL VOC 2011 validation images. For each row, we show the input image, ground-truth and the segmentation results of FCN-8s along side the intermediate outputs of our LRR-4x model at the 32x, 16x and 8x layers.
References

1. Shotton, J., Winn, J., Rother, C., Criminisi, A.: Textonboost for image understanding: Multi-class object recognition and segmentation by jointly modeling texture, layout, and context. IJCV 81(1) (2009) 2–23
2. Carreira, J., Sminchisescu, C.: Cpmc: Automatic object segmentation using constrained parametric min-cuts. PAMI 34(7) (2012) 1312–1328
3. Carreira, J., Li, F., Sminchisescu, C.: Object recognition by sequential figure-ground ranking. IJCV 98(3) (2012) 243–262
4. Hariharan, B., Arbeláez, P., Girshick, R., Malik, J.: Hypercolumns for object segmentation and fine-grained localization. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. (2015) 447–456
5. Long, J., Shelhamer, E., Darrell, T.: Fully convolutional networks for semantic segmentation. In: CVPR. (2015) 3431–3440
6. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. (2015)
7. Burt, P.J., Adelson, E.H.: The laplacian pyramid as a compact image code. Communications, IEEE Transactions on 31(4) (1983) 532–540
8. Dai, J., He, K., Sun, J.: Convolutional feature masking for joint object and stuff segmentation. (3 December 2014)
9. Mostajabi, M., Yadollahpour, P., Shakharovich, G.: Feedforward semantic segmentation with zoom-out features. In: CVPR. (2015) 3376–3385
10. Kokkinos, I.: Pushing the boundaries of boundary detection using deep learning. In: ICLR. (2016)
11. Chen, L.C., Barron, J.T., Papandreou, G., Murphy, K., Yuille, A.L.: Semantic image segmentation with task-specific edge detection using cnns and a discriminatively trained domain transform. (2016)
12. Chen, L.C., Papandreou, G., Kokkinos, I., Murphy, K., Yuille, A.L.: Semantic image segmentation with deep convolutional nets and fully connected crfs. In: ICLR. (2015)
13. Lin, G., Shen, C., Hengel, A.v.d., Reid, I.: Efficient piecewise training of deep structured models for semantic segmentation. In: CVPR. (2016)
14. Liu, Z., Li, X., Luo, P., Loy, C.C., Tang, X.: Semantic image segmentation via deep parsing network. In: ICCV. (2015) 1377–1385
15. Koltun, V.: Efficient inference in fully connected crfs with gaussian edge potentials. Adv. Neural Inf. Process. Syst (2011)
16. Zheng, S., Jayasumana, S., Romera-Paredes, B., Vineet, V., Su, Z., Du, D., Huang, C., Torr, P.H.: Conditional random fields as recurrent neural networks. In: ICCV. (2015) 1529–1537
17. Zeiler, M.D., Taylor, G.W., Fergus, R.: Adaptive deconvolutional networks for mid and high level feature learning. In: Computer Vision (ICCV), 2011 IEEE International Conference on, IEEE (2011) 2018–2025
18. Zeiler, M.D., Fergus, R.: Visualizing and understanding convolutional networks. In: Computer vision–ECCV 2014. Springer (2014) 818–833
19. Noh, H., Hong, S., Han, B.: Learning deconvolution network for semantic segmentation. In: ICCV. (2015) 1520–1528
20. Xie, S., Tu, Z.: Holistically-nested edge detection. In: Proceedings of the IEEE International Conference on Computer Vision. (2015) 1395–1403
21. Yang, S., Ramanan, D.: Multi-scale recognition with dag-cnns. In: ICCV. (2015) 1215–1223
22. Dai, J., He, K., Sun, J.: Boxsup: Exploiting bounding boxes to supervise convolutional networks for semantic segmentation. In: ICCV. (2015) 1635–1643
23. Chen, L.C., Yang, Y., Wang, J., Xu, W., Yuille, A.L.: Attention to scale: Scale-aware semantic image segmentation. arXiv preprint arXiv:1511.03339 (2015)
24. Gidaris, S., Komodakis, N.: Object detection via a multi-region and semantic segmentation-aware cnn model. In: ICCV. (2015) 1134–1142
25. Denton, E.L., Chintala, S., Fergus, R., et al.: Deep generative image models using a laplacian pyramid of adversarial networks. In: Advances in Neural Information Processing Systems. (2015) 1486–1494
26. Lee, C.Y., Xie, S., Gallagher, P., Zhang, Z., Tu, Z.: Deeply-supervised nets. (2015)
27. Hariharan, B., Arbeláez, P., Bourdev, L., Maji, S., Malik, J.: Semantic contours from inverse detectors. In: ICCV. (2011) 991–998
28. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition. arXiv preprint arXiv:1409.1556 (2014)
29. Vedaldi, A., Lenc, K.: Matconvnet – convolutional neural networks for matlab
30. Kohli, P., Torr, P.H., et al.: Robust higher order potentials for enforcing label consistency. International Journal of Computer Vision 82(3) (2009) 302–324
31. Lin, T.Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollár, P., Zitnick, C.L.: Microsoft coco: Common objects in context. In: ECCV. (2014) 740–755