LEARNING SHAPE PRIORS FOR OBJECT SEGMENTATION VIA NEURAL NETWORKS

Simon Safar *Ming-Hsuan Yang*

University of California at Merced

ABSTRACT

We present a joint algorithm for object segmentation that integrates both global shape and local edge information in a deep learning framework. The proposed architecture uses convolutional layers to extract image features, followed by a fully connected section to represent shapes specific to a given object class. This preliminary mask is further refined by matching segmentation mask patches to local features. These processing steps facilitate learning the shape priors effectively with a feedforward pass rather than complex inference methods. Furthermore, our novel convolutional refinement stage presents a convincing alternative to Conditional Random Fields, with promising results on multiple datasets.

***Index Terms*—** Object segmentation, shape priors, convolutional neural networks

1. INTRODUCTION

Generating per-pixel foreground-background masks for objects is a challenging task with many applications in computer vision and image processing. In this paper, we focus on the segmentation task itself: given a bounding box and an object class (e.g., horse, human or bird), we aim to infer the mask within the bounding box itself. In this process, we make use of not only local cues but also global shape characteristics of a specific object class, thereby making our approach robust against significant amount of occlusion.

In this work, we propose an algorithm to learn effective shape priors using deep neural networks. Instead of relying on hand-crafted features, we use the features from a pre-trained deep neural network on the ImageNet dataset [1]. In addition, we learn the connection between shapes, pixel masks and image features in a fully supervised way.

The proposed algorithm consists of two modules: image analysis and mask synthesis. The analysis module of the network processes raw visual information from the object of interest. These features are used by the synthesis module to determine the overall shape of the object and and to refine its pixel-wise segmentation mask.

Typically, nodes in the earlier layers of the analysis module represent simpler visual features, such as edges or colors, while the ones in later layers indicate the presence of more complicated structures, such as human faces or legs of horses. However, this presents a tradeoff as by collecting visual information from larger receptive fields, layers in the later layers lose location information that would be essential for accurate segmentation. In contrast to the feature extraction stage, our synthesis network uses the opposite architecture. First, a coarse, global mask is inferred using inputs from the later, more high-level layers of the analysis network. This is then used as one of the inputs for the more fine-grained second layer, providing the final output. For this step, we use the early-stage visual information from the analysis side, to better localize parts of the mask.

The contributions of this work are twofold. First, we show that learning shape priors does not require complicated probabilistic inference. Given the effective features, a feedforward layer can be efficiently applied for this purpose. Second, we present an algorithm for refining segmentation masks using convolutional networks. To the best of our knowledge, we are first to propose such an approach.

2. RELATED WORK AND PROBLEM CONTEXT

For object segmentation, we can make use of not only local appearance but also global shape information for effective inference. One approach to represent global shape is to use a Conditional Random Field (CRF) with a Potts model (based on pairwise potentials) to learn foreground-background masks directly from pixels [2]. This model does not require learning numerous parameters and the inference can be performed easily using an efficient Graph Cut algorithm [3]. The method based on Compositional High Order Pattern Potential (CHOPP) [4] extends the pairwise model by hidden nodes connected to all mask pixels, and learns global correlations between them. Eslami et al. [5] use the Restricted Boltzmann Machine with two hidden layers to learn a shape model. Recently, the Max-Margin Boltzmann Machine (MMBM) algorithm [6] adds a direct connection from the image features towards the hidden nodes to make their estimates more robust against spurious local predictions. In this work, we only use this direct connection towards the hidden nodes to identify shapes. This enables us to infer the correct output even with noisy inputs and still keep the simple, feedforward inference.

Aside from pixel-based representations, object shapes can also be represented by patches and matched. In [7], Dong

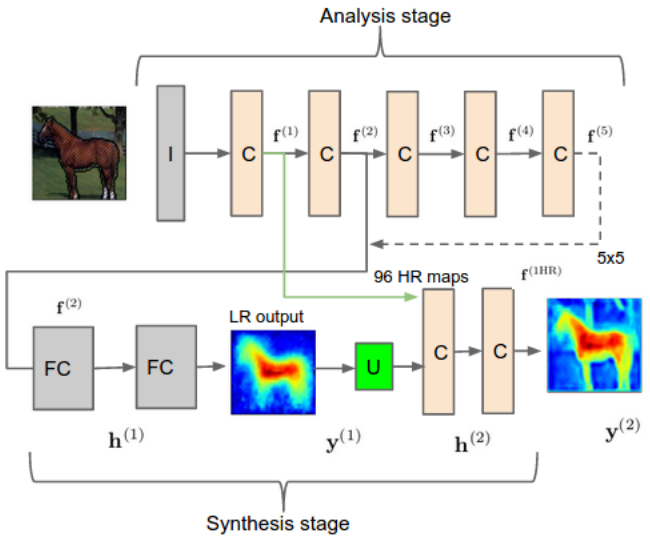

Fig. 1. Architecture of our segmentation network, with the image analysis stage on the top and the mask synthesis stage at the bottom where C, U, and FC denote a convolutional layer, an upsampling operation, and a fully connected layer, respectively. LR and HR stand for "Low Resolution" and "High Resolution".

et al. introduce a deep learning network that formulates the patch matching process in the form of convolutional and fully connected layers. In this paper, we exploit both representations to learn shape priors for effective object segmentation.

3. NETWORK ARCHITECTURE

3.1. Shapes

The first stage of our network is similar in spirit to the MMBM1 model [6] where the feature vector $\mathbf{x}$ is obtained based on the Histogram of Oriented Gradients (HOG), along with color and shape histograms, and then fed to a fully connected layer. Both the features and hidden nodes are connected to an output raster for generating the binary output mask. However, as demonstrated by our first experiments, it is not necessary to apply a similarly complex inference scheme. Using a backpropagation-based, stochastic gradient descent training process instead is a viable and valuable alternative, with the trade-off of a minor decrease in accuracy against significantly shorter training times as well as simpler implementation.

As shown in Fig. 1, the inputs of our synthesis network are obtained from the convolutional outputs of the analysis stage. Given an image $\mathbf{I}$, we feed it to the five pre-trained convolutional layers. We extract features using the pre-trained deep network [1], resulting in the analysis output $\mathbf{f}^{(1)}$, ..., $\mathbf{f}^{(5)}$, from each layer respectively (e.g., if there was a normalization component, we take its output; if not, we use the pooling output instead). We note these feature detectors are generic as the deep network is trained based on the ImageNet dataset with a large variety of objects with demonstrated success.

The pre-trained deep network performs pooling after the first layer and generates the feature map $\mathbf{f}^{(1)}$ with half the resolution of the eventual output mask. Thus, we also perform the same pooling operation with a stride of 1 instead of 2. To compensate for the change of resolution, local response normalization is carried out on this result with a kernel size of 9 rather than 5 to generate the higher-resolution feature maps $\mathbf{f}^{(1\text{HR})}$.

The first half of the synthesis network alone does not exploit other visual cues that are useful to describe object shapes. With this network, the global shape information is encoded in the hidden nodes $\mathbf{h}^1$. Since this representation is only capable of describing global shape information, it does not account for small local variations in shape, thereby resulting in the blurry output mask $\mathbf{y}^1$ as shown in the second row of Fig. 2.

3.2. Patches

To better describe object shapes with more visual cues, we learn a mapping between image features and segmentation masks both globally and locally, and associate different patterns of output mask patches with image features. We first describe them with local linear feature maps, and then with a nonlinear second layer to predict the output center pixel at a specific location. Although our approach is conceptually similar to the recent deep super resolution method [7], the goal is to refine the global shape prior with local correction, not learning the segmentation mask only from the image patches.

We formulate the mask refinement stage as a composition of two convolution steps. The first step uses a 5×5 kernel to collect local shape information, followed by Rectified Linear Units (ReLU) and a local response normalization layer. The operation for the k-th output feature map can be summarized as a layer-wise convolution with a filter bank F,

$$\mathbf{h}_k^{(2)} = \max(\mathbf{0}, \sum_j \{\mathbf{f}^{(1)}, \mathbf{y}^{(1\text{HR})}\}_j * F_j) \circ \frac{1}{Z_k} \qquad (1)$$

in a similar operation to the analysis layers, on the concatenation of the prior $\mathbf{f}^{(1)}$ and the high-resolution feature maps $\mathbf{y}^{(HR)}$. The local response normalization term within the neighborhood $\mathcal{N}(x, y)$ of the pixel at (x, y) is

$$Z_k(x, y) = (1 + (\alpha/n) \sum_{i \in \mathcal{N}(x,y)} s_{k,i}^2)^\beta, \qquad (2)$$

where $\alpha = 0.0001$ and $\beta = 0.75$ are parameters, $n = 25$ is the number of pixels in the receptive field and the s_k-s are the un-normalized feature maps from the left term of (1). We extract the feature maps $\mathbf{h}^{(2)}$ using (1). These are further processed by a linear per-pixel transformation, described by a 1×1 convolution, providing the refined mask output $\mathbf{y}^{(2)}$.

The use of local patch representation alone is not sufficient to learn object shape priors well. If we use only local features, the resulting network may learn local edges but not the global context and object shape. Thus, in order to combine local edge information with a global shape prior, we connect not only the image feature maps $\mathbf{f}^{(1)}$ but also an upsampled version of $\mathbf{y}^{(1)}$ of the prior generated by our first layer to the patch predictors in (1). This allows the refinement process to discard locally promising regions that do not fit the predicted shape while still predicting local edge shapes more accurately.

The second row of Fig. 2 shows the shape prior $\mathbf{y}^{(1)}$ which localizes the object and successfully predicts its overall shape, but produces blurry predictions regarding the exact boundaries. This is successfully corrected by the patch match component, producing the output $\mathbf{y}^{(2)}$.

In practice, we generate the shape prior with a smaller resolution than that of $\mathbf{y}^{(2)}$ which is upsampled between the two stages, denoted by U on the figure. Since the shape prior has low spatial accuracy due to its global nature, this allows us to decrease the number of parameters in the fully connected layers without reducing accuracy while speeding up the process.

4. EXPERIMENTS

4.1. Learning deep networks

Implementation details. In this work, we use the Caffe deep learning framework [8] and extend it with additional layer types (in-place reshape and upsample) that are not available in the package. Most of the processing is carried out on an nVidia GeForce GTX Titan GPU, with the most important constraint being the amount of available memory (memory usage being on the order of 2 GB in the training stage). Also, we are using templates [9] to generate models for the different stages and datasets, to avoid code repetition. All the source code and datasets will be made available to the public.

Features. To extract features, we use a pre-trained deep network [1] using the ImageNet dataset in which we re-learn the biases of the first convolutional layer to account for issues between different application domains (i.e., cross-domain problems). For training the first shape prior stage, we experiment with different choices of input features and find that the number of pre-trained image processing layers does not affect the resulting accuracy significantly, given a sufficient amount of training iterations. Based on these experiments, we use the locally normalized output $\mathbf{f}^{(2)}$ of the second convolutional layer as the input for our shape prior. For the second convolution layer, the normalized first convolutional output $\mathbf{f}^{(1HR)}$ is used in order to obtain the highest spatial resolution for accurate segmentation.

Training process. Although training the two components of the synthesis network jointly also gives good results, we find that the training process is more stable if the loss of the first global stage is assigned significantly more weight during the backpropagation process than the loss of the second layer. Inasmuch, we train the network layer-wise, obtaining performance gains on the order of a few percents.

Further improvements can be achieved by considering the different purposes of the first and second layers. For the global shape prior, in the case of uncertainty, the network can be better trained with accurate probability estimates using the logistic loss function

$$L_{\text{global}}(\mathbf{y}^{(1)}, \mathbf{y}^{(\text{gt})}) = \sum_i -\log(\mathbf{1}_{y_i^{(1)}=y_i^{(\text{gt})}}\sigma(y_i^{(1)}) + \mathbf{1}_{y_i^{(1)}\neq y_i^{(\text{gt})}}(1-\sigma(y_i^{(1)}))) \quad (3)$$

where $y_i^{(\text{gt})} \in \{-1, 1\}$ are the ground truth mask pixels.

In contrast, the loss function for the final output $\mathbf{y}^{(2)}$ approximates the pixelwise accuracy by

$$L_{\text{local}}(\mathbf{y}^{(2)}, \mathbf{y}^{(\text{gt})}) = ||\sigma(\mathbf{y}^{(2)}) - \mathbf{y}^{'(\text{gt})}||_2 \quad (4)$$

where $\mathbf{y}^{'\text{gt}} = \frac{\mathbf{y}^{\text{gt}}+1}{2}$ is the ground truth scaled to a range of 0 to 1. These loss functions are used in training the first and second components of the synthesis network, respectively. For the proposed network, we use 500 hidden nodes for $\mathbf{h}^{(1)}$ and 30 feature maps for $\mathbf{h}^{(2)}$.

4.2. Datasets

To validate our model, we use the Weizmann Horses [10] and Caltech-UCSD Birds 200 [11] datasets where the pixel-wise ground-truth segmentation mask of each object is manually annotated. The Weizmann Horses dataset contains 328 images with large variation of shape, pose, and scale. The Caltech-UCSD Birds dataset includes 6033 images of 200 species where each one appears in different shapes, sizes, poses with occlusions by tree branches. For fair comparisons, we use the same experimental setup as [6] to pre-process and split the training as well as test samples for both datasets. Due to space limitations, additional results (including those on the Penn-Fudan Pedestrians dataset [4]) are available at `http://eng.ucmerced.edu/people/ssafar/icip15_shapepriors_supp.pdf`.

4.3. Results

We evaluate the proposed algorithm against the state-of-the-art methods in terms of average precision (AP) and intersection over union (IoU) measures. Table 1 and 2 show the experimental results where for [6], the Graph Cut (GC) method is used to further improve performance. Some segmentation results are presented in Fig. 2. Overall, the proposed algorithm performs favorably against other methods. For the Weizmann Horses dataset, large shape variation (e.g., caused by leg position and body pose) can be better accounted for with the two-layer network of the MMBM method over single-layer

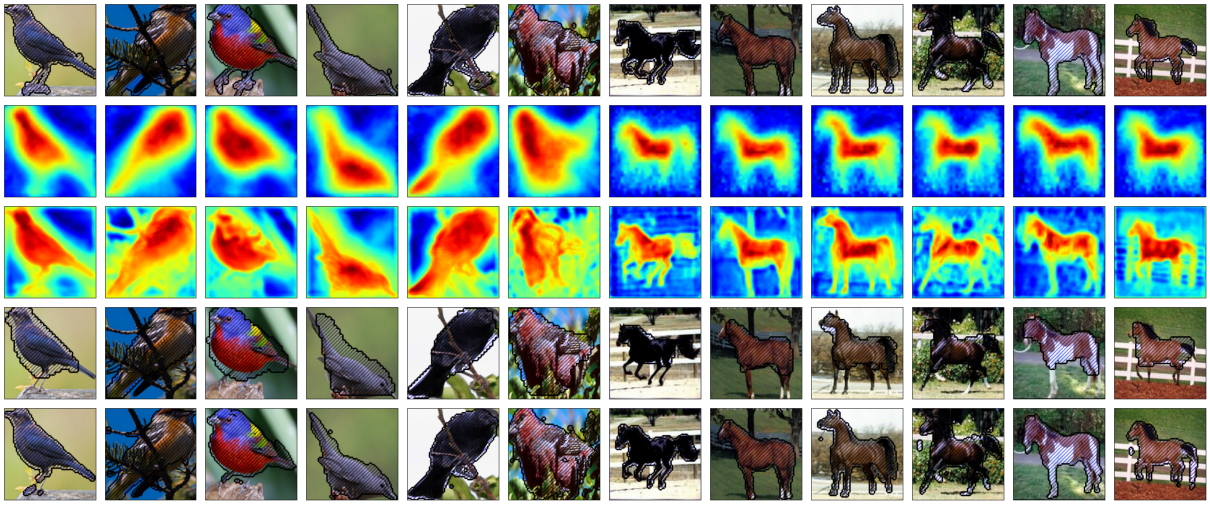

Fig. 2. Example results from the convolutional refinement on the Caltech-UCSD Birds and Weizmann Horses datasets. From top to bottom: ground truth, global shape prior $\mathbf{y}^{(1)}$, refined map $\mathbf{y}^{(2)}$, global prior mask, refined mask. On the maps, colors nearing red correspond to higher probabilities of a pixel being foreground.

Table 1. Results on the Weizmann Horses dataset.

	AP	IoU
CRF [6]	87.46	67.44
CHOPPs [4]	88.67	71.60
MMBM (1 layer) [6]	89.43	69.59
MMBM (2 layers)	89.80	72.09
MMBM (1 layer), with GC	90.62	74.12
MMBM (2 layers), with GC	90.71	75.78
Ours, global only	83.88	49.96
Ours, refined	**95.16**	**83.55**

Table 2. Results on the Caltech-UCSD Birds 200 dataset.

	AP	IoU
CRF [6]	83.50	38.45
CHOPPs [4]	74.52	48.84
MMBM (1 layer) [6]	88.07	72.96
MMBM (2 layers)	86.38	69.87
MMBM (1 layer), with GC	90.42	75.92
MMBM (2 layers), with GC	**90.77**	72.40
Ours, global only	84.02	69.40
Ours, refined	88.27	**76.30**

MMBM approach (74.12% IoU for the MMBM1 with GC, 69.59% for MMBM1 without GC as shown in Table 1). By using patchwise refinement, our method is able to follow not only the main body contours but also the outline of the legs accurately, thereby resulting in significant accuracy increase on this dataset.

The last column of Fig. 2 shows some cases where the proposed method does not perform well. The overall shape prior $\mathbf{y}^{(1)}$ constrains where horse legs are likely to appear (under torso), and then the second stage of the learned network model discovers their exact locations. As the white fence post on the right of the input image is locally similar to a leg in shape and appearance at the likely position with respect to the torso, it is mistakenly classified as part of the foreground object. On the other hand, the similar fence post on the left is mostly classified properly as background thanks to the context given by the global shape prior. Further results on the interplay between the two stages can be found at the above-mentioned web page.

The Caltech-UCSD Birds 200 dataset includes a wider range of shapes and highly variable colors. In terms of overlap ratio, the first global segmentation stage of the learned network model performs similarly to the two-layer MMBM model. However, while the graph cut refinement is only able to achieve 2.96% IoU increase in the one-layer model and 2.53% for the two-layer network, our convolutional refinement gains 6.90% over the initial global mask with the best overlap score of all methods. In terms of average precision, the graph cut refinement still performs well. In addition, our method consistently outperforms pure CRF inference and shows significant gains over CHOPP, which demonstrates the importance of shape priors and a direct connection of these to image features.

5. CONCLUSION

In this paper, we describe a two-stage deep neural network for object segmentation that exploits learned features and shape priors. Our algorithm is simple to implement and insensitive to the network parameters (e.g., number of layers). It performs favorably against the state-of-the-art object segmentation methods on challenging datasets.

6. REFERENCES

[1] A. Krizhevsky, S. Ilya, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *NIPS*, 2012.

[2] J. Lafferty, A. McCallum, and F. Pereira, "Conditional random fields: Probabilistic models for segmenting and labeling sequence data," in *ICML*, 2001.

[3] Y. Boykov and V. Kolmogorov, "An experimental comparison of min-cut/max-flow algorithms for energy minimization in vision," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2004.

[4] Y. Li, D. Tarlow, and R. Zemel, "Exploring compositional high order pattern potentials for structured output learning," in *CVPR*, 2013.

[5] S. M. A. Eslami, N. Heess, and J. Winn, "The shape Boltzmann machine: a strong model of object shape," in *CVPR*, 2012.

[6] Jimei Yang, Simon Safar, and Ming-Hsuan Yang, "Max-margin boltzmann machines for object segmentation," in *CVPR*, 2014.

[7] C. Dong, C. Loy, K. He, and X. Tang, "Learning a deep convolutional network for image super-resolution," in *ECCV*, 2014.

[8] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, and T. Darrell, "Caffe: Convolutional architecture for fast feature embedding," *arXiv preprint arXiv:1408.5093*, 2014.

[9] R. Koebler, "Pyratemp templating framework," `http://www.simple-is-better.org/template/pyratemp.html`.

[10] E. Borenstein and S. Ullman, "Class-specific, top-down segmentation," in *ECCV*, 2002.

[11] P. Welinder, S. Branson, T. Mita, C. Wah, F. Schroff, S. Belongie, and P. Perona, "Caltech-UCSD Birds 200," Tech. Rep., California Institute of Technology, 2010.