Learning Regularization Weight for CRF Optimization

Jiaxiao Wu
http://www.csd.uwo.ca/
University of Western Ontario
1151 Richmond St
London, ON
Canada

Olga Veksler
https://cs.uwaterloo.ca/
University of Waterloo
200 University Ave W
Waterloo, ON
Canada

Abstract

CNNs have made a tremendous impact on the field of computer vision in the last several years. For dense image classification tasks, CNNs usually assign a label to each pixel individually. To enforce spatial coherence, CNNs are often combined with CRFs. CRF optimization includes a regularization term that helps to obtain a spatially consistent labeling. The weight of the regularization term, usually learned from the training data, is fixed to the same value for all images. However, for optimal results, it is best to use a different amount of regularization for each image depending on the image content and the reliability of the label probabilities learned by CNN. We propose to learn the regularization weight from training data for each individual image. To this end, we construct a dataset where the optimal regularization weight for CRF optimization has been precomputed for each image. We use a CNN with a standard architecture, but the input is tailored to our problem. The input is the image itself and the label probabilities produced by CNN, since the regularization weight depends on both. The output of our CNN is the regularization weight, which is then used with a CRF optimizer in a post processing step. We test the effectiveness of our approach on the task of salient object segmentation. We show that accuracy improvement can be achieved when using regularization weight learned on per-image basis as opposed to using a single regularization weight learned for all images in the dataset.

1 Introduction

Convolutional Neural Networks (CNNs) [12, 17, 20] have led to tremendous successes in computer vision in the past several years, initially excelling in image classification tasks [17, 20]. Fully convolutional neural networks [25] proposed an efficient way to apply CNNs to the problem of semantic segmentation, a task where each image pixel needs to be assigned a label. Now CNNs are being used for a variety of pixel labeling problems, such as semantic segmentation [10, 25], stereo correspondence [6, 34], optical flow [9], etc.

CNNs label each image pixel individually based on the estimated label probabilities. Due to the overlap in the receptive fields, the label probabilities produced by CNN at nearby pixels

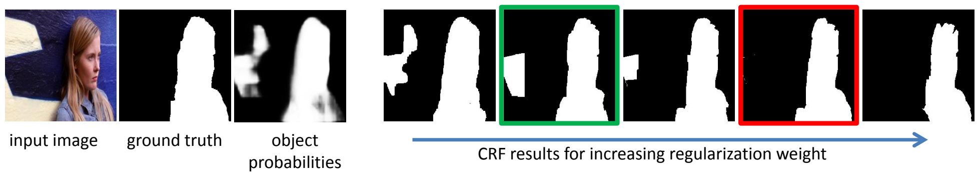


Figure 1: The three leftmost images show the input image, the ground truth, and salient object probabilities produced by a trained CNN. The rest of the images are the CRF segmentation results as the regularization weight is increased. The segmentation outlined in green is obtained with the learned regularization weight that has been fixed to the same value for the whole dataset. The segmentation outlined in red is obtained with the regularization weight estimated by our trained CNN for this particular image, based on image content and salient object probabilities.

are correlated. However, the label dependencies are not explicitly modeled. Therefore the results can exhibit a degree of spatial incoherence. Conditional Random Fields (CRFs) [16, 19] were designed specifically for the task of modeling pixel label dependencies. To improve the coherence and therefore the accuracy of pixel labeling, CNNs are often combined with CRFs. CRFs are either added as a post processing step [4, 10], or constitute a part of an end-to-end trainable system [5, 14, 36].

When combining CRFs with CNNs, the probabilities learned by the CNN are converted to the unary CRF terms, and then a learned or fixed pairwise CRF terms are added. The pairwise terms impose some type of regularization. The result tends to be more spatially coherent compared to using the unary terms alone, leading to an improved labeling accuracy. Regularization (pairwise) terms are added with a weight for controlling the strength of the regularization. The regularization weight is either set by hand or learned through the training data, or a combination of set-by-hand and learned parameters. In the prior work that combines CNNs with CRFs, while the pairwise terms can depend on the content of an individual image, such as edge contrast, the overall regularization weight is learned as a single parameter with a fixed value for all database images.

For each individual image, however, the best result is achieved if the regularization weight is adapted to that specific image, instead of using a single regularization weight that works best for all the images in the dataset. This is because the optimal amount of regularization required for a particular image depends on the image content, not just edge contrast, and also on the quality of the unary terms produced by CNN for that particular image. If the unary terms are noisy, the regularization weight should be increased. Assuming length regularization, if there are finely structured objects in the image, regularization weight should be decreased. There are other non obvious relationships between the best regularization weight and the image content and CNN unary terms that one can attempt to learn.

In this paper, we propose to learn the optimal regularization weight from the image content and the unary terms. To learn the regularization weight, we use the CNN with a standard architecture, referred to as *weight-CNN*. The input to weight-CNN is the color image and the unary terms used in the CNN-CRF system, since the optimal regularization weight depends on both. The output is the regularization weight for this particular image which we then use during optimization in a CNN-CRF system.

Consider Figure 1. The first three images show the input image, the ground truth, and salient object probabilities produced by the CNN. The rest of the images show segmentation results with an increasing regularization weight obtained by a CNN-CRF system. If one uses the regularization weight that works best for all the images in the dataset, that is the best fixed regularization weight, then the result outlined in green is obtained. If one uses our proposed weight-CNN for learning the optimal regularization weight for each individual image, then the result outlined in red is obtained. Notice that in this case, the salient object probability map is rather noisy, and the object does not have fine structures. Therefore the learned regularization weight for this particular image has a larger value than the fixed regularization weight learned for the whole dataset.

We chose a simple application of salient object segmentation where the task is to segment a salient object [7] from its background. We chose this application, in part, because an efficient and globally optimal CRF optimizer is available in this case [1]. We use a simple CNN-CRF approach, where the CRF is used as a post-processing step. We use the CNN from [21] to learn unary terms for our CNN-CRF system. We also use the same CNN to produce the input required by our weight-CNN. The overview of our proposed framework is in Figure 2.

Since there is no prior work on learning the regularization weight in a supervised manner, we first construct a training dataset. The examples are the color images coupled with the unary terms learned by CNN from [21]. The labels are the optimal regularization weight, estimated empirically from the ground truth saliency segmentations. We show that with our learned regularization weight, we can improve the accuracy of CNN-CRF system, using the standard F-measure metric used in saliency segmentation.

This paper is organized as follows. In Section 2 we discuss the prior work, in Section 3 we give details of our overall CNN-CRF system, in Section 4 we describe how we construct our dataset, in Section 5 we give the architecture of our CNN for learning regularization strength, in Section 6 we present the experimental results. Finally, conclusions are in Section 7.

2 Prior Work

We are not aware of any prior work that uses deep learning for estimating regularization weight for CRF optimization. However there is prior work on parameter learning for CRFs, and prior work on combining CRFs with CNNs. We review these in this section.

First we discuss the prior work on learning CRF parameters. One approach to CRF parameter learning is to use the classical method of maximum likelihood estimation [18, 19, 22]. In the context of CRFs, though, maximum likelihood estimation is intractable and various approximations, such as pseudo-likelihood are used instead. These construct local approximations instead of true global distributions, and have no quality guarantees. Instead of using local approximations, in [27] they propose using approximate global distributions based on the most-probable-explanation principle [8] with the hope of obtaining more reliable estimates. Still, a single regularization weight is estimated for the whole dataset, as opposed to on per-image basis, which is our goal.

In [35], they learn CRF parameters using statistical estimation based on expectation maximization (EM) in the context of the stereo correspondence application. Their approach does learn parameters per image. However, it is computationally intensive, as the proposed EM procedure has to be run for multiple iterations.

There are also methods for learning CRF parameters in the context of structured learning [29, 30]. These methods can be efficient provided an efficient CRF optimizer is available. However, they learn a fixed regularization weight parameter for all images in the dataset.

In [26], they use an empirical approach to choosing the best regularization weight for each image individually. First they train an AdaBoost [11] classifier that assigns a "goodness" score to a segmentation. Given a new image to segment, they run CRF optimization for different regularization weight values and apply the trained classifier to each segmentation. They chose the segmentation with the best "goodness" score as their final result. While interesting, this approach is costly since CRF optimization has to be performed multiple times and the classifier for the segmentation result has to be run several times as well.

We now turn to reviewing the approaches for combining CNNs with CRFs. The simplest and earliest approaches apply CRF in a post processing step [4, 10]. The probabilities for class labels learned by a CNN are converted into the unary CRF terms. Then a standard CRF optimization algorithm, such as in [3] or [16] is used to obtain the final labeling. This approach is simple but does not support end-to-end training.

In order to support end-to-end training, some methods implement a CRF optimizer inside the neural network. In [36], they implement mean-field inference [15], a popular optimizer for fully-connected CRFs, as a Recurrent Neural Network (RNN). This method is limited to mean field inference, which has been shown an ineffective optimizer [31].

The methods in [5, 14] use a structural learning framework. The unary and/or pairwise CRF terms are modeled with CNN-computed features. This approach supports end-to-end training, but the loss function is difficult to optimize and various approximations have to be used.

3 CNN-CRF System

We use a simple CNN-CRF combined system, where a CRF is used to post process CNN results. Our system is outlined in Figure 2. We use the CNN from [21] to learn saliency probabilities, which we refer to as "saliency-CNN". The input to saliency-CNN is a color image, and the output is the probability of each pixel being salient. The probability map from saliency-CNN and the input image serve as an input into weight-CNN. The probability map is concatenated as the fourth channel to the color image. The output of weight-CNN is the regularization strength which we name λ, see Eq. (2). The input image, saliency probabilities, and lambda are the input to the CRF, which then produces the final segmentation. The CRF optimizes the energy in Eq. (1), explained below.

We use the binary (two labels) energy function proposed in [1] for segmenting a salient object from its background. Let $\mathcal{P}$ be all the image pixels, and $x_p \in \{0,1\}$ the label assigned to pixel p. Here $x_p = 0$ means pixel p is assigned to the background, $x_p = 1$ means pixel p is assigned to the salient object. Let $x = (x_p \,|\, p \in \mathcal{P})$ be the vector of all pixel assignments. The CRF energy is defined as

$$f(x) = \sum_{p \in \mathcal{P}} f_p(x_p) + \sum_{p,q \in \mathcal{P}} w_{pq} \cdot [x_p \neq x_q]. \tag{1}$$

The unary term $f_p(x_p)$ is the cost of assigning pixel p to label x_p. It is modeled as negative log probability of the salient (if $x_p = 1$) or background (if $x_p = 0$) object. These probabilities are obtained by training saliency-CNN. The pairwise term $w_{pq} \cdot [x_p \neq x_q]$ is the penalty one pays whenever pixels p, q are not assigned to the same label. We set w_{pq} to be inversely

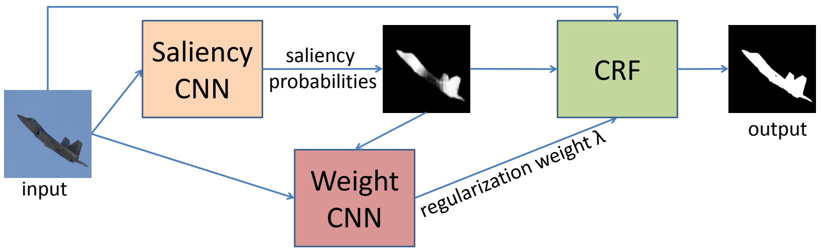


Figure 2: Our CNN-CRF combined system, see text for description.

proportional to the distance between the color vectors of pixels p and q, weighted by the regularization weight parameter λ

$$w_{pq} = \lambda \cdot exp\left(-\frac{||C_p - C_q||^2}{2\sigma^2}\right). \tag{2}$$

We use a standard 4-connected CRF. This CRF energy regularizes a labeling by encouraging a shorter boundary length. The larger the value of λ, the stronger the regularization influence in comparison to the unary terms. The main goal of this paper is to learn how to set the best value for parameter λ, tailored to each individual image. The optimum labeling minimizing the energy in Eq. (1) can be found efficiently via the minimum cut algorithm [2].

4 Dataset Construction

Since there is no prior work on learning the regularization weight λ in Eq. (2) in a supervised manner, we first construct a training dataset. The value of λ controls the regularization strength and the smoothness of the segmentation boundaries. An appropriate λ value encourages boundary smoothness and suppresses noise caused by inaccurate unary terms. However, the details on the object boundary might be lost if λ is too large. On the contrary, if the λ value is too small, too much noise remains in the final segmentation. Thus, for each image, the value of λ in Eq. (2) that gives the best visual result is usually different. We need to determine this best λ value as the correct label or 'ground truth' for each image in our dataset. Notice that our training dataset is specific to the energy and the unary terms in Eq. (1). The training examples are the color images coupled with the unary terms learned by the CNN, and the correct labels are the optimal λ's.

We start with the standard salient segmentation dataset [24]. For each image, we use the CNN from [21] to obtain a pixelwise probability map for the salient object. The saliency probability map is converted to the unary terms for Eq. (1) by taking negative logarithm.

Next, for each image i in the dataset, we need to know the value of λ in Eq. (2) that gives the best F-measure on image i, which we refer to as λ_i^*. We compute the values of λ_i^* empirically. That is, for each image i, we perform optimization of the energy in Eq. (1) for a range of λ values. For each segmentation result, we compute the F-measure, which is a standard performance measure on the saliency benchmarks. Finally, we set the λ value giving the best F-measure to be λ_i^*. We call λ_i^* for image i as the 'ground truth λ'. In

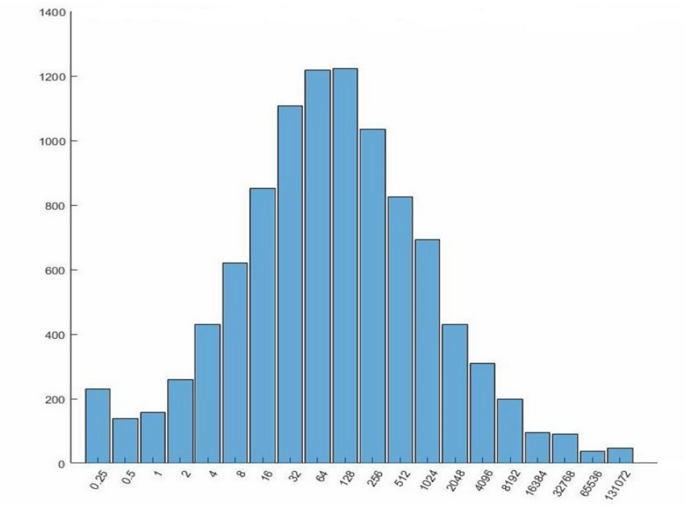


Figure 3: The number of images assigned to each possible value of λ^* for MSRA10K dataset

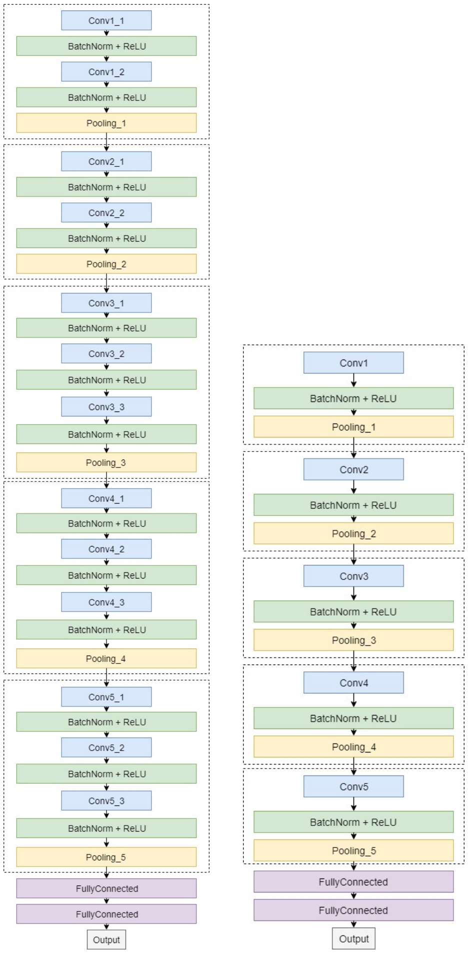
order to limit the search space, we fix the set of λ values to explore to be $L = \{2^i |\ i \in \{-2,-1,...,17\}\}$. We call set L the λ-spectrum.

We choose the lower and upper limits of λ-spectrum empirically. We observe that λ^* rarely drops below 2^{-2} and rarely rises above 2^{17}. We decide that the interval between each value of λ in our spectrum should grow exponentially because the *relative* error seems to affect our results more than *absolute* error. For example, the difference between $\lambda = 10$ vs $\lambda = 11$ seems to matter far more than that of $\lambda = 10000$ vs $\lambda = 10001$. The distribution of empirically estimated λ^* values computed for the MSRA10K dataset [24] is shown in Figure 3.

5 Weight-CNN Architecture

In this section, we explain the architecture of weight-CNN for learning the optimal regularization weight λ for each image. We exploit two regression network architectures that are both alterations of the VGG-16 nets [28]. We adopt the main features of the VGG-16 network structure because VGG-16 and its variations have shown a remarkable performance in various vision tasks other than object classification for which VGG-16 was developed. Their recent success in semantic segmentation demonstrated the expressive power and the potential of the VGG-16 structure to solve other vision tasks.

We modify the VGG-16 net to our weight-CNN by reducing the number of neurons in the output layer from the number of classes (for classification purposes) to one which outputs a real scalar value serving as the exponent of the predicted regularization weight value. We take the number of neurons in the fully connected layers as a hyperparameter to tune during training. We preserve the 5 max pooling layers with stride 2 in the original VGG-16 net. This is to reduce the feature map resolution in the hope of extracting more compact, positionally invariant feature representations. Each convolutional layer has 3×3 kernels with stride 1,



(a) Full-WCNN (b) Trimmed-WCNN

Figure 4: The architecture of regression networks.

and the number of kernels grows exponentially between each convolutional group as in the VGG nets. Zero padding is employed before every convolutional layer for easy calculation of changes in feature map dimension.

One difference between our two networks is the number of convolutional layers between each pooling layer. As shown in Figure 4 (a), the full weight-CNN network (Full-WCNN) has two to three convolutional layers in each stage. A stage contains one or more convolutional layers, followed by a batch normalization layer, ReLU activation layer and a max pooling layer. Meanwhile, for the trimmed weight-CNN network (Trimmed-WCNN) in Figure 4 (b), there is only one convolutional layer in each stage. Additionally, in Full-WCNN, the two fully connected layers have the same number of neurons; meanwhile, in Trimmed-WCNN, the second fully connected layer has half as many neurons as in the first fully connected layer. A single neuron is attached after the fully connected layers to generate a real number output, which is used to compute the predicted regularization weight.

Trimmed-WCNN is slimmer than Full-WCNN in terms of the number of trainable weights, which forces Trimmed-WCNN to learn more efficiently and significantly reduces the potential overfitting problem. The slimness also reduces training time. Holding the other configurations consistent, Trimmed-WCNN networks generally needs two-thirds as much time to train as Full-WCNN.

The networks are trained on the mean squared logarithmic error (MSLE) calculated on the ground truth values of λ^* and the network predictions. Let λ_i^* be the ground truth label for image i and let $f_\theta(x_i)$ be the raw network output. Then we use the following loss function on the batch of n images

$$L_{MSLE} = \frac{1}{n}\sum_{i=1}^{n}(log(\lambda_i^*) - log(f_\theta(x_i)))^2. \tag{3}$$

We choose the mean squared logarithmic error instead of mean squared error because we intend to penalize the relative error instead of the absolute error, especially considering that our λ^* values are obtained from a search with exponential increment.

6 Experimental Results

We use the MSRA10K [24] saliency dataset for weight-CNN training. It provides pixel-level saliency labeling for 10,000 images, and is one of the largest saliency datasets with pixel precise ground truth.

We augment the MSRA10K dataset to include the per-image based optimal regularization weight. All color images are resized to size 256 by 256 using bilinear interpolation, and the ground truth images are resized to the same dimensions using nearest neighbor interpolation. Then, we randomly partition the augmented MSRA10K dataset into the training set, validation set and test set, and each containing $8,000$, $1,000$ and $1,000$ images respectively. During the training stage, the mini-batch size is set to 20. We trained weight-CNN for 100 epochs and took the network at a checkpoint that resulted in the best F-measure on validation data. This usually occurred before 50 epochs were reached. The networks were trained with the Adam [13] optimizer to minimize a loss function consisting of MSLE loss and L_2 weight decay with a decay factor of 5×10^{-4}. Learning rate was set to 10^{-3}.

We focus our result analysis on F-measure, a standard evaluation metric for saliency segmentation, which is defined as

$$F_\beta = (1+\beta^2) \times \frac{precision \times recall}{(\beta^2 \times recall) + precision}, \tag{4}$$

where we set $\beta^2 = 0.3$, as standard. The precision and recall are calculated as $\frac{t_p}{t_p+f_p}$ and $\frac{t_p}{t_p+f_n}$ respectively, where t_p, f_p, and f_n stand for true positives, false positives, and false negatives respectively.

We now compare the performance of CNN-CRF system using the per-image λ learned with weight-CNN and performance of CNN-CRF system using λ fixed for the whole dataset. We call λ fixed for the whole dataset λ_{fixed}. To find λ_{fixed}, we go over all possible values in L, defined in Section 4, and for each value compute the average F-measure on the training dataset. Then we choose λ_{fixed} as the value maximizing the average F-measure. For the MSRA10K dataset, $\lambda_{fixed} = 64$. Table 1 demonstrates the comparison results. The first

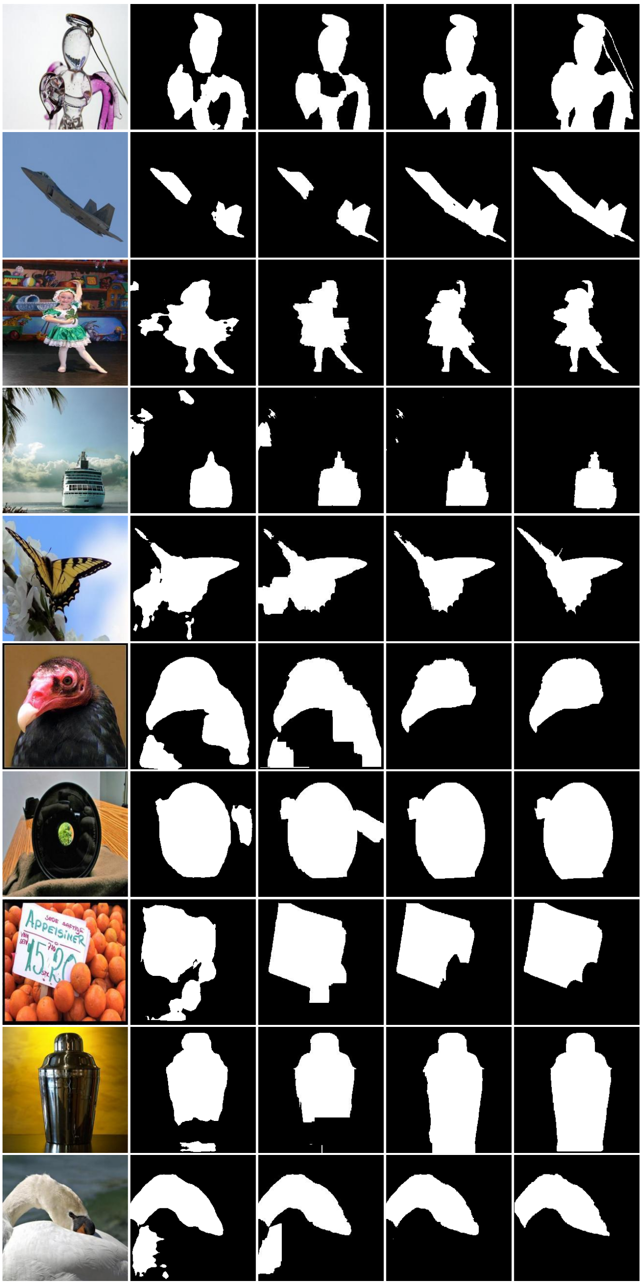


Figure 5: Some example results. From left to right: original image, segmentation produced by saliency-CNN, segmentation of CNN+CRF with the same regularization weight λ for all images, segmentation of CNN+CRF with λ learned by our weight-CNN, ground truth.

three lines give the performances with a per-image chosen variable λ, and the last line with λ_{fixed}. The first line is the result with λ computed from the ground truth, i.e. the F-measure corresponding to the optimal λ^* computed per image. Thus the results with per-image learned (or fixed) λ cannot exceed the first line. The second line is the F-measure using per-image learned λ with the Full-WCNN network, and the third line is the F-measure using the Trimmed-WCNN network. With a per-image learned λ, we show almost 1% improvement compared to using λ_{fixed}. However, there is still quite some room for improvement, since with the ground truth λ^* the F-measure is still higher. Learning the optimal λ is far from a trivial task. Figure 5 shows some sample results.

Method	Selected with	F-measure
Variable λ	Ground truth	0.94391557
Variable λ	Full-WCNN	0.91660978
Variable λ	Trimmed-WCNN	0.91550659
Fixed λ	$\lambda_{fixed} = 64$	0.90786851

Table 1: F-measures of different λ acquisition methods evaluated on the MSRA10K test set

An effective learning model should have steady performance on different benchmark datasets. To assess the effectiveness of the proposed weight learning approach, we evaluate the weight-CNN model, which was trained with the MSRA10K dataset, on another three contemporary saliency segmentation datasets, including PASCAL [23], ECSSD [32], and DUT-OMRON [33]. As shown in Table 2, consistent improvements of various scales are observed on all investigated public available saliency benchmarks.

Method	Selected with	F-measure		
		PASCAL	ECSSD	DUT-OMRON
Variable λ	Ground truth	0.85261991	0.92567453	0.76539855
Variable λ	Full-WCNN	0.79429388	0.89112503	0.68927901
Variable λ	Trimmed-WCNN	0.79748042	0.89188917	0.68416296
Fixed λ	$\lambda_{fixed} = 64$	0.79403781	0.88792895	0.67819961

Table 2: F-measures of different λ acquisition methods evaluated on the PASCAL, ECSSD, and DUT-OMRON datasets

7 Conclusions

We propose a novel deep learning approach to obtain a per-image based regularization weight for a CNN-CRF system, and apply the CNN-CRF system with the learned regularization weight to the problem of salient object segmentation. For learning the regularization weight, we design a convolutional regression network that takes an input color image augmented by the probabilities used for unary terms in CRF. This is because regularization weight depends not just on the image content, but also on the quality of the unary terms. We show that with our per-image learned regularization weight we obtain better results compared to using the best fixed regularization weight for all the images in the dataset. In the future, we plan to explore learning regularization weight in an end-to-end CNN+CRF system.

References

[1] Yuri Boykov and Gareth Funka-Lea. Graph cuts and efficient nd image segmentation. *International journal of computer vision*, 70(2):109–131, 2006.

[2] Yuri Boykov and Vladimir Kolmogorov. An experimental comparison of min-cut/max-flow algorithms for energy minimization in vision. *IEEE transactions on pattern analysis and machine intelligence*, 26(9):1124–1137, 2004.

[3] Yuri Boykov, Olga Veksler, and Ramin Zabih. Fast approximate energy minimization via graph cuts. *IEEE Transactions on pattern analysis and machine intelligence*, 23 (11):1222–1239, 2001.

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

[5] Liang-Chieh Chen, Alexander Schwing, Alan Yuille, and Raquel Urtasun. Learning deep structured models. In *International Conference on Machine Learning*, pages 1785–1794, 2015.

[6] Zhuoyuan Chen, Xun Sun, Liang Wang, Yinan Yu, and Chang Huang. A deep visual correspondence embedding model for stereo matching costs. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 972–980, 2015.

[7] Ming-Ming Cheng, Niloy J Mitra, Xiaolei Huang, Philip HS Torr, and Shi-Min Hu. Global contrast based salient region detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 37(3):569–582, 2015.

[8] Robert G Cowell, Philip Dawid, Steffen L Lauritzen, and David J Spiegelhalter. *Probabilistic networks and expert systems: Exact computational methods for Bayesian networks*. Springer Science & Business Media, 2006.

[9] Alexey Dosovitskiy, Philipp Fischer, Eddy Ilg, Philip Hausser, Caner Hazirbas, Vladimir Golkov, Patrick Van Der Smagt, Daniel Cremers, and Thomas Brox. Flownet: Learning optical flow with convolutional networks. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2758–2766, 2015.

[10] Clement Farabet, Camille Couprie, Laurent Najman, and Yann LeCun. Learning hierarchical features for scene labeling. *IEEE transactions on pattern analysis and machine intelligence*, 35(8):1915–1929, 2013.

[11] Yoav Freund and Robert E Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of computer and system sciences*, 55 (1):119–139, 1997.

[12] Kunihiko Fukushima. Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position. *Biological Cybernetics*, 36(4):193–202, Apr 1980. ISSN 1432-0770.

[13] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[14] Patrick Knöbelreiter, Christian Reinbacher, Alexander Shekhovtsov, and Thomas Pock. End-to-end training of hybrid cnn-crf models for stereo. In *Conference on Computer Vision and Pattern Recognition,(CVPR)*, 2017.

[15] Daphne Koller and Nir Friedman. *Probabilistic graphical models: principles and techniques*. MIT press, 2009.

[16] Philipp Krähenbühl and Vladlen Koltun. Efficient inference in fully connected crfs with gaussian edge potentials. In *Advances in neural information processing systems*, pages 109–117, 2011.

[17] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.

[18] Sanjiv Kumar and Martial Hebert. Man-made structure detection in natural images using a causal multiscale random field. In *null*, page 119. IEEE, 2003.

[19] John Lafferty, Andrew McCallum, and Fernando Pereira. Conditional random fields. In *ICML*, 2001.

[20] Yann LeCun, Bernhard Boser, John S Denker, Donnie Henderson, Richard E Howard, Wayne Hubbard, and Lawrence D Jackel. Backpropagation applied to handwritten zip code recognition. *Neural computation*, 1(4):541–551, 1989.

[21] Guanbin Li and Yizhou Yu. Deep contrast learning for salient object detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 478–487, 2016.

[22] Stan Z Li. Markov random field models in computer vision. In *European conference on computer vision*, pages 361–370. Springer, 1994.

[23] Yin Li, Xiaodi Hou, Christof Koch, James M Rehg, and Alan L Yuille. The secrets of salient object segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 280–287, 2014.

[24] Tie Liu, Jian Sun, Nan-Ning Zheng, Xiaoou Tang, and Heung-Yeung Shum. Learning to detect a salient object. In *Computer Vision and Pattern Recognition, 2007. CVPR'07. IEEE Conference on*, pages 1–8. IEEE, 2007.

[25] Jonathan Long, Evan Shelhamer, and Trevor Darrell. Fully convolutional networks for semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3431–3440, 2015.

[26] Bo Peng and Olga Veksler. Parameter selection for graph cut based image segmentation. In *BMVC*, volume 32, pages 42–44, 2008.

[27] Daniel Scharstein and Chris Pal. Learning conditional random fields for stereo. In *Computer Vision and Pattern Recognition, 2007. CVPR'07. IEEE Conference on*, pages 1–8. IEEE, 2007.

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

[29] Martin Szummer, Pushmeet Kohli, and Derek Hoiem. Learning crfs using graph cuts. In *European conference on computer vision*, pages 582–595. Springer, 2008.

[30] Ben Taskar, Vassil Chatalbashev, Daphne Koller, and Carlos Guestrin. Learning structured prediction models: A large margin approach. In *Proceedings of the 22nd international conference on Machine learning*, pages 896–903. ACM, 2005.

[31] Yair Weiss. Comparing the mean field method and belief propagation for approximate inference in mrfs. *Advanced mean field methods: theory and practice*, pages 229–240, 2001.

[32] Qiong Yan, Li Xu, Jianping Shi, and Jiaya Jia. Hierarchical saliency detection. In *Computer Vision and Pattern Recognition (CVPR), 2013 IEEE Conference on*, pages 1155–1162. IEEE, 2013.

[33] Chuan Yang, Lihe Zhang, Huchuan Lu, Xiang Ruan, and Ming-Hsuan Yang. Saliency detection via graph-based manifold ranking. In *Computer Vision and Pattern Recognition (CVPR), 2013 IEEE Conference on*, pages 3166–3173. IEEE, 2013.

[34] Jure Zbontar and Yann LeCun. Computing the stereo matching cost with a convolutional neural network. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1592–1599, 2015.

[35] Li Zhang and Steven M Seitz. Estimating optimal parameters for mrf stereo from a single image pair. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 29(2):331–342, 2007.

[36] Shuai Zheng, Sadeep Jayasumana, Bernardino Romera-Paredes, Vibhav Vineet, Zhizhong Su, Dalong Du, Chang Huang, and Philip HS Torr. Conditional random fields as recurrent neural networks. In *Proceedings of the IEEE international conference on computer vision*, pages 1529–1537, 2015.