Instance-sensitive Image Segmentation with Mask-RCNN and CRF

Haodong Zhou
Dept. of Computer Science
haodoz1@uci.edu

Tzu-Chi Lin
Dept. of Computer Science
tzucl2@uci.edu

Abstract

Image understanding problems such as object detection and semantic segmentation have made breakthroughs in recent years due to the development of deep learning techniques. We aim to provide a method to tackle with an instance segmentation problem that can be done in pixel level. Our model used Mask-RCNN incorporates with a conditional random field to further refine the result to the pixel level[4].

1 Introduction

Instance segmentation is one of a fundamental problem in computer vision that combines object detection and semantic segmentation. Semantic segmentation labels every pixel to its object class, without distinguishing different objects. Object detection does localize the object in a course, bounding-box level. These two problems are well-studied topics in scene understanding and have recently risen due to deep learning. The rising of autonomous driving and robotics industry also made this problem more important, since the accurate recognition is one of the main tasks in these two industries.

There are two different approaches to instance segmentation. First, based on object detection pipelines, where objects are first localized to different boxes, then applies the semantic segmentation method to further refine the object in each box. Another result is based on segment-based but rather box-based method. However, these methods do not consider the whole image, but rather independent proposals. Moreover, since they process numerous proposal independently, it is hard to produce segmentation maps of the image.

We provide a model which is based on the fully convolution neural network, and corporate the conditional random field to refine the result. Fully convolution neural network has been proved an efficient model to segmentation problem. We use CRF as a post-processing method, which has successfully pursued a joint learning method with CNN. We aim to use mean-field inference to convert the system into an end-to-end network, so we can use the convolutional layer as features to approximate CRF mean field inference.

2 Related work

Convolution Networks are driving advanced in segmentation. Traditional convolution networks solve for whole-image tasks like classification, and for local tasks like bounding box object detection. Because of the lack of the position information of the last layer in the network, it's hard to make a pixel level prediction of an image.

The fully convolutional network (FCN) trains an end-to-end, pixels-to-pixels network, which does a pixelwise prediction from supervised pre-trainning [4] . To improve the result of the convolution network, a conditional random field (CRF) is used to improve the marginal performance by Chen, L. C. et al [1]. Chen, L. C. [2, 3] claims the performance of semantic segmentation can be improved

Submitted to 32nd Conference on Neural Information Processing Systems (NIPS 2018). Do not distribute.

Figure 1: Workflow of Instance Segmentation System

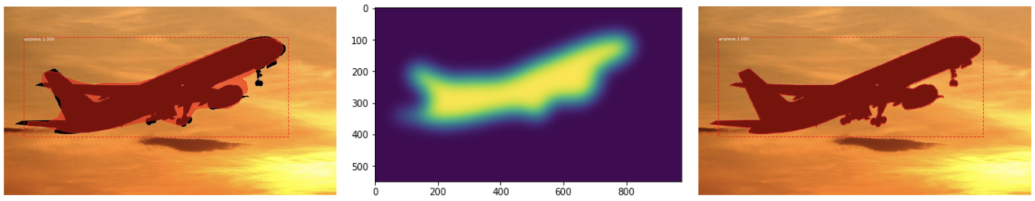

Figure 2: Left: result from Mask-RCNN. Middle: result generated by smoothing the mask, as the input to the CRF, which helps to improve the border performance. Right: result from the CRF refinement.

by using the atrous spatial pyramid pooling (ASPP) and using batch normalization within each ASPP. Also, an Encoder-Decoder with astrous convolution model was introduced to improve the performance.

However, these works mainly focus on image semantic segmentation. For instance-sensitive segmentation, Dai, J. et al [5] proposed a method that uses different score map to distinguish different instance. Li, Y. et al [6] helps to improve the performance by training a region proposal network (RPN) in the model, which proposes region-of-interests on the score map for joint object segmentation and detection.

3 Proposed approach

3.1 Pipeline

Figure 1 and 2 shows a simple pipeline of our model. We aim to combine Mask-RCNN and CRF to get a refined result. Given an image as input, firstly we use Mask-RCNN to generate a bounding box and a mask for the image. Considering that the border of the mask is improvable, we use a Gaussian kernel to smooth the mask, which makes the border of the mask to be the state of "improvable" while keeps the center of the mask a high probability value.

The probability value generated by the smoothing kernel is then consider the unary potential of the CRF model. We further define the pairwise potential function as describe in section 3.3 and use mean field approximation to update maximum a posteriori probability (MAP) iteratively as described in section 3.4.

3.2 Mask-RCNN

Mask-RCNN is a kind of network that achieve three functions: it can generate a bounding box, showing the position of the object, generate a class number and a possibility value indicating the type of the object and the probability, and the mask of the object. The Mask-rcnn shows a good result in the instance segmentation task.

The Faster-RCNN [8] is a network that can do multi-object detection and classification. The basic work flow of the Faster-RCNN is going through a CNN, using a RPN branch to generate ROIs(Region of Interest), sending the ROIs into a ROI Pooling layer and a classifier to get the bounding box and a classification result.

The Mask-RCNN [7] is based on the Faster-RCNN. It uses a FCN branch to generate the mask of the instance. Also, it uses the ROI Align layer instead of the ROI Pooling, which solves the problem caused by same feature size generated by different size ROIs. This will cause a low precision of the

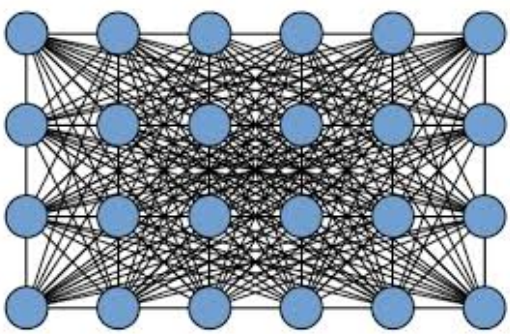

Figure 3: Example of fully connected CRF

mask. Mask-RCNN is a instance sensitive segmentation, and it has the state-of-the-art result, which is used as a baseline in our project.

3.3 Fully Connected Conditional Random Field Model

We use the fully connected CRF model to further refine the result, which is shown in Figure 3. We can see each pixel is represented by each node in CRF model. Consider a random field **X** defined over a set of variables $\{X_1, ..., X_n\}$ which ranges over possible pixel-level image labeling. Suppose there are k label, then the domain of each variable in X is a set of labels $L = \{l_1, ...l_k\}$. Consider another random field **I** defined over $\{I_1, ..., I_n\}$, which indicates the color vector of the pixel.

Now we have two conditional random fields to consider, joint these two CRFs to get a new CRF (**X**,**I**), which can be represented by Gibbs distribution $P(X|I) = \frac{1}{Z(I)} exp(-\sum \phi_c(X_c|I))$. Also, the Gibbs energy can be expressed as $E(x|I) = \sum \phi_c(x_c|I)$, which as our unary potential function in fully connected CRF. The maximum a-posterior (MAP) is $x^* = argmax_{x \in L^N} P(x|I)$, which is the distribution we want to compute.

The Gibbs energy in the fully connected CRF model can be represented as

$$E(x) = \sum_i \psi_u(x_i) + \sum_{i \neq j} \psi_p(x_i, x_j)$$

The unary potential $\psi_u(x_i)$ is computed independently for each pixel by mask-RCNN. Since the unary potential is produced independently, we could consider pairwise potential $\psi_p(x_i, x_j)$ to make the image more consistent. The pairwise potential has the form

$$\psi_p(x_i, x_j) = \mu(x_i, x_j) \sum_{m=1}^{K} w^{(m)} k^{(m)}(f_i, f_j)$$

where μ is the label compatibility function, $k^{(m)}(f_i, f_j)$ is a Gaussian kernel and $w^{(m)}$ is its weight of linear combination. We could express the kernel function as two Gaussian kernel, considered in both color vectors and position

$$k(f_i, f_j) = w^{(1)} exp(-\frac{|p_i - p_j|^2}{2\theta_\alpha^2} - \frac{|I_i - I_j|^2}{2\theta_\beta^2}) + w^{(2)} exp(-\frac{|p_i - p_j|^2}{2\theta_\gamma^2})$$

where **I** indicates color vector and **p** is the position of the pixel. The first Gaussian kernel is called *appearance kernel*, which is inspired by the nature of the object - nearby pixels with similar color are likely to be the same object. Later kernel is called *smoothness kernel*, which aims to remove isolated region.

3.4 Mean Field Approximation

The algorithm to compute MAP is based on the mean field approximation to the CRF distribution. We could see this approximation as a message passing algorithm and run it iteratively to get the approximation inference.

Assume the KL-divergence we want to compute is $D(Q||P)$. We can derive this Q as

$$Q_i(x_i = l) = \frac{1}{z_i} exp\{-\psi_\mu(x_i) - \sum_{l' \in L} \mu(l, l') \sum_{m=1}^{K} w^{(m)} \sum_{j \neq i} k^{(m)}(f_i, f_j) Q_j(l')\}$$

A detailed derivation is given in Appendix. We can decompose this formula into three steps. First, We can see $\sum_{m=1}^{K} w^{(m)} \sum_{j \neq i} k^{(m)}(f_i, f_j) Q_j(l')$ as a message passing from all X_i to X_j. $\sum_{l' \in L} \mu(l, l')$ is a compatibility transformation. And we update $exp\{-\psi_\mu(x_i) - Q'\}$ locally at last step.

For this message passing step, it takes quadratic time. We can improve this step to linear time by the observation this message passing step could be represented as a Gaussian filter in feature space [9]. We then only need to do a sampling and performs low-pass filter on this sample, which can be performed in O(N) time.

4 Implementation Details

Our project is based on the tensorflow. The Mask-RCNN part we used the implementation of the matterport, and the model is a pre-trained model [1], which already performs a good result. The output we get from the Mask-RCNN is a group of bounding box with a binary mask inside.

Then we apply CRF to masks one by one. We use a adaptive size Gaussian kernel to smooth the mask. Then the result will be sent into the CRF function.

We adaptive our code to denseCRF code [2]. Since we already have the unary potential from Mask-RCNN, we only have to create two Gaussian kernels for pairwise potential. There are some parameters we could tune on. For smoothness kernel parameter θ_γ, it does not affect the result significantly. We just set $\theta_\gamma = 1$ and found it work well. Unfortunately, the appearance kernel parameters θ_α and θ_β cannot be computed effectively by gradient decent, since their gradient involves a sum of non-Gaussian kernels, which are not amenable to the same acceleration techniques. We simply use a grid search to find θ_α and θ_β.

After setting all the parameters in kernels, we need to run inference algorithm to compute MAP and assign each pixel by MAP. These steps could all be done by denseCRF API. Since we already found out 10 iterations may converge for KL-divergence as described in section 5.1, we set all other experiments to run 10 times iterations.

5 Experiments

5.1 Convergence of mean field approximation

We test the convergence of the mean field approximation first by analyzing KL-divergence between Q and P. Figure 4 shows the KL-divergence over iterations of the inference algorithm. We could see that it converges after 10 iterations. Hence, we set 10 as the number of iterations in all other experiments.

5.2 Grid search for parameters

Figure 6 and 7 shows a simple result for grid search of parameters θ_α and θ_β. We can see that for $\theta_\alpha = 60$ and $\theta_\beta = 10$ is good for our result. We then set $\theta_\alpha = 60$ and $\theta_\beta = 10$ to the following experiment.

5.3 Evaluation method

The performance is evaluated by intersection-over-union (IOU) between the predicted instances and the ground-truth instances. We report the standard COCO metrics including AP (averaged over IoU thresholds), AP_{50}, AP_{75} , and AP_S , AP_M , AP_L (AP at different scales), which is the same method with the Mask-RCNN

[1] github.com/matterport/Mask_RCNN/

[2] github.com/lucasb-eyer/pydensecrf/

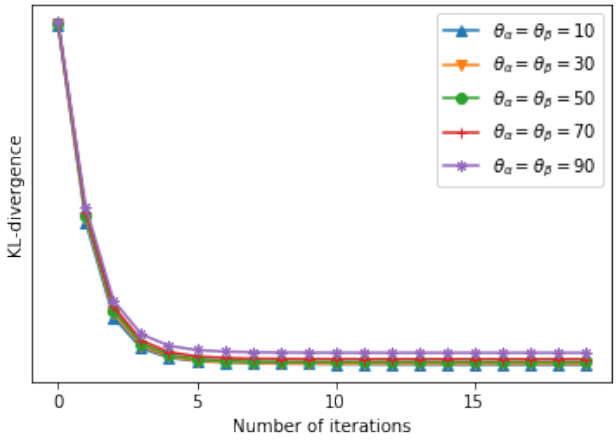

Figure 4: Convergence of KL-divergence

Figure 5: Iteration result of the image

(a) θ_α=1 (b) θ_α=20 (c) θ_α=40 (d) θ_α=60 (e) θ_α=80

Figure 6: Grid search for θ_α

(a) θ_β=1 (b) θ_β=10 (c) θ_β=20 (d) θ_β=30 (e) θ_β=40

Figure 7: Grid search for θ_β

Table 1: Instance Segmentation Result on COCO2014 Dataset

method	AP	AP_{50}	AP_{75}	AP_S	AP_M	AP_L
Mask R-CNN	35.7	58.0	37.8	15.5	38.1	52.4
Mask R-CNN with CRF	35.2	58.7	36.4	17.3	36.5	49.8

5.4 Result and Analysis

The result based on the evaluation method is in the Table 1. Our result overall is similar with the result of the Mask R-CNN. Parameters is pretty hard to automatically adapt for each mask, and the current result is based on the parameters which makes small objects performs better than original Mask-RCNN.

Some other results with specified tuned parameters are showed in the Figure 8, 9 and 10 . We can find that after 10 iterations of CRF, the border of the mask is improved. However, to achieve these result, we increase the weight of the color, which causes that the part of the bird which has the similar color with the background color to has poor performance.

6 Future Work

To further improve the performance, we may consider to put CRF model to end-to-end model so that we could jointly optimize the parameters of the CNN and the CRF, such as CRF as RNN model[10].

7 Appendix

7.1 Detailed Derivation of Q

We give a detailed derivation of Q in this section.

First we have Gibbs distribution of denseCRF

$$P(X) = \frac{1}{Z}P'(X) = \frac{1}{Z}exp(\sum_i \psi_u(x_i) + \sum_{i \neq j} \psi_p(x_i, x_j))$$

We can derive KL-divergence as following

$D(Q||P) = \sum_x Q(x)log(\frac{Q(x)}{P(x)}) = -\sum_x Q(x)logP(x) + \sum Q(x)logQ(x)$
$= -E_{X \in Q}[logP(X)] + E_{X \in Q}[logQ(x)]$
$= -E_{X \in Q}[logP'(X)] + E_{X \in Q}[logZ] + \sum_i E_{X_i \in Q}[logQ_i(X_i)]$
$= -E_{X \in Q}[logP'(X)] + logZ + \sum_i E_{X_i \in Q_i}[logQ_i(X_i)]$

Since our goal is to compute Q and $logZ$ do not have Q, we ignore this term.

At the same time, Q have to satisfy

$$\sum_{x_i} Q_i x_i = 1$$

By Lagrange Multiplier, we can get

$$L(Q_i) = -E_{X \in Q}[logP'(X)] + \sum_i E_{X_i \in Q_i}[logQ_i(X_i)] + \lambda(\sum_{x_i} Q_i(x_i) - 1)$$

By partial differential we can get

$$\frac{\partial L(Q_i)}{\partial Q_i(x_i)} = -E_{\bar{X} \in Q_i}[logP'(X|x_i)] - logQ_i(x_i) - 1 + \lambda$$

Set this to 0, we can get

$$Q_i(x_i) = exp(\lambda - 1)exp(-E_{\bar{X} \in Q_i}[logP'(X|x_i)])$$

We can see $exp(\lambda - 1)$ as a constant number, so Q becomes

$$Q_i(x_i) = \frac{1}{Z_i} exp(-E_{\bar{X} \in Q_i}[logP'(X|x_i)])$$

where Z is a normalization term.

We put P' to this formula as we define at the beginning.

$$Q_i(x_i) = \frac{1}{Z_i} exp(-E_{\bar{X} \in Q_i}[\sum_i \psi_u(x_i) + \sum_{i \neq j} \psi_p(x_i, x_j)|x_i])$$

Since x_i is known as label l, we now have

$Q_i(x_i = l) = \frac{1}{Z_i} exp(-\psi_\mu(l) - \sum_{j \neq i} E_{\bar{X} \in Q_j} \psi_p(l, X_j))$

$= \frac{1}{Z_i} exp(-\psi_\mu(l) - \sum_{m=1}^{K} w^{(m)} \sum_{j \neq i} E_{X \in Q_j}[\mu(l, X_j) k^{(m)}(f_i, f_j)])$

$= \frac{1}{Z_i} exp(-\psi_\mu(l) - \sum_{j \neq i} \sum_{l' \in L} k^{(m)}(f_i, f_j) \mu(l, l') Q_j(l'))$

$= \frac{1}{z_i} exp\{-\psi_\mu(x_i) - \sum_{l' \in L} \mu(l, l') \sum_{m=1}^{K} w^{(m)} \sum_{j \neq i} k^{(m)}(f_i, f_j) Q_j(l')\}$

where we can the result of

$$Q_i(x_i = l) = \frac{1}{z_i} exp\{-\psi_\mu(x_i) - \sum_{l' \in L} \mu(l, l') \sum_{m=1}^{K} w^{(m)} \sum_{j \neq i} k^{(m)}(f_i, f_j) Q_j(l')\}$$

7.2 Contribution of each member

Haodong Zhou is mainly responsible for Mask-RCNN, including its description in this paper and implementation of the code. Also, section 2 is written by him.

Tzu-Chi Lin is mainly responsible for CRF model, including its description in this paper and implementation of the code. Also, section 1 is written by him.

We together figure out how to combine Mask-RCNN and CRF together.

References

[1] Chen, L. C., Papandreou, G., Kokkinos, I., Murphy, K., & Yuille, A. L. (2018) Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. *IEEE transactions on pattern analysis and machine intelligence*, **40**(4), pp.834-848.

[2] Chen, L.C., Papandreou, G., Schroff, F., & Adam, H. (2017) Rethinking atrous convolution for semantic image segmentation. *arXiv preprint*, arXiv:1706.05587.

[3] Chen, L. C., Zhu, Y., Papandreou, G., Schroff, F., & Adam, H. (2017) Encoder-decoder with atrous separable convolution for semantic image segmentation. *arXiv preprint* arXiv:1802.02611.

[4] Long, Jonathan, Evan Shelhamer, & Trevor Darrell. (2015) Fully convolutional networks for semantic segmentation. *Proceedings of the IEEE conference on computer vision and pattern recognition* pp.3431-3440.

[5] Dai, J., He, K., Li, Y., Ren, S., & Sun, J. (2016) Instance-sensitive fully convolutional networks. *European Conference on Computer Vision* pp. 534-549.

[6] Li, Y., Qi, H., Dai, J., Ji, X., & Wei, Y. (2016) Fully convolutional instance-aware semantic segmentation. *arXiv preprint* arXiv:1611.07709.

[7] He, K., Gkioxari, G., Dollár, P. & Girshick, R., 2017, October. Mask r-cnn. *In Computer Vision (ICCV), 2017 IEEE International Conference on* (pp. 2980-2988). IEEE.

[8] Ren, S., He, K., Girshick, R., & Sun, J., (2015). Faster r-cnn: Towards real-time object detection with region proposal networks. *In Advances in neural information processing systems* (pp. 91-99).

[9] Krahenbuhl, P. and Koltun, V. (2011) Efficient inference in fully connected crfs with gaussian edge potentials. In NIPS.

[10] Zheng, S., Jayasumana, S., Romera-Paredes, B., Vineet, V., Su, Z., Du, D., Huang, C., and Torr, P. (2015) Conditional random fields as recurrent neural networks, in IEEE ICCV.

Figure 8: Example With Tuned Parameters

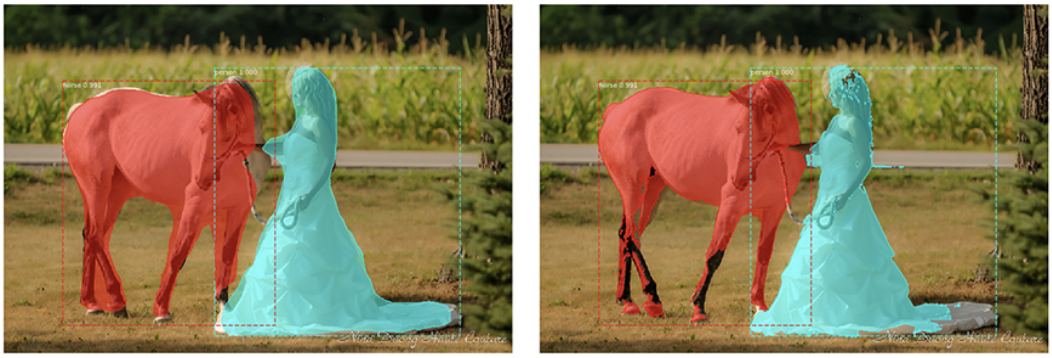

Figure 9: Example With Tuned Parameters

Figure 10: Example With Tuned Parameters