Using Deep Features for Segmentation Annotations

Sabine Süsstrunk – Teacher
sabine.susstrunk@epfl.ch

Baran Ozaydin – Assistant
baran.ozaydin@epfl.ch

Nicolas Gonzalez – Student
nicolas.gonzalez@epfl.ch

Abstract

Pixel-wise annotations are crucial for the segmentation task but they require extensive human effort and time. Our aim is to incorporate deep features in the annotation process and ease the annotation cost/time. We're soliciting a pre-trained convolutional neural network – SqueezeNet [3], in order to access class activation maps [9], which, are in turn used by a standard foreground extraction algorithm – GrabCut [6]. Tested on the the PASCAL VOC - 2012 dataset, our technique achieves a mIoU of 68.5 for instance segmentation (slight improvement to the 67.7 mIoU achieved by naked GrabCut).

1. Introduction

Computer vision heavily relies on image segmentation and annotations. These two tasks are usually computationally expensive and/or require significant human interventions. Managing to improve them could be very valuable, considering the importance that the computer vision field has taken in the recent decades.

Studies [8, 9] have suggested that convolutional neural network may act as object localizer. Also, as shown in [4,5, 10] CNN are really well suited for image segmentation.

Our plan was to combine the CNN tools with classical foreground extraction algorithms. Our technique relies on class activation maps [9] and GrabCut [6]. The transformation of the class activation map into a GrabCut input mask constitutes the central idea of our algorithm.
Tested on the the PASCAL VOC - 2012 dataset, our technique achieves **34.9** mIoU for class segmentation, and **68.5** mIoU for instance segmentation. The hybrid GrabCut-CAM technique performed slightly better than GrabCut, or CAM on their own.

The next two sections aim to provide a theoretical overview of the two major components of our GrabCut-CAM algorithm.

2. Class activation maps

Class activation mapping is a convolutional neural networks related concept brought by Zhou *et al*. [9].

> A class activation map for a particular category indicates the discriminative image regions used by the CNN to identify that category. [9]

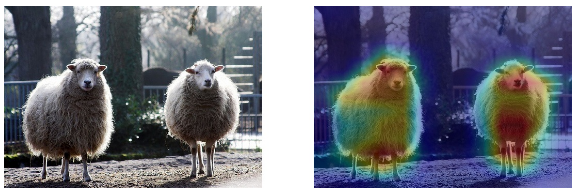

Figure 1. CAM for class *Ram*. Red shades indicates a strong discriminative effect.

One key idea originating from paper [9], is that CAMs can be outputted with only very minor architecture changes to an image classification – trained CNN.
Concretely, multiplying the output of the last convolutional layer by the class score weights, before proceeding to global average pooling, allows one to obtain the CAMs.
Let's have a quick mathematical overview of the situation.

Assume that we've fed to our network an image for evaluation. Using [9]'s terminology and notations, let c be some class and

- S_c be the input to the softmax for class c,
- w_k^c be the weights for class c and unit k,
- $f_k(x, y)$ be the activation of unit k in the last convolutional layer at spatial location (x, y) and
- $M_c(x, y)$ be the class activation map for class c at spatial location (x, y).

Let us then consider the following set of equations

$$S_c = \sum_k w_k^c \sum_{x,y} f_k(x,y) \tag{1}$$

$$= \sum_{x,y} \sum_k w_k^c f_k(x,y) \tag{2}$$

$$= \sum_{x,y} M_c(x,y). \tag{3}$$

In principle, in order to compute the class score S_c, one would first perform global average pooling (GAP) with $\sum_{x,y} f_k(x,y)$, and then only do the units – linear combination, as show in equation (1). By swapping the GAP and linear combination, as shown in equations (2), it is possible to bring out the term $\sum_k w_k^c f_k(x,y)$, written as $M_c(x,y)$ in (3) *i.e.* the class activation map for class c. Although computationally more expansive, this allows one to compute the class score and get the CAM in one pass.

3. GrabCut

“GrabCut” is an interactive foreground extraction algorithm elaborated by Rother *et al.* [6] which itself relies on a graph cut technique proposed by Boykov and and Jolly [1].

3.1. GrabCut masks

GrabCut takes as input an image and a mask and returns another mask. Masks depict the foreground region to be extracted. They take value in the $\{Foreground, Background\} \times \{Probable, Definite\}$ space.

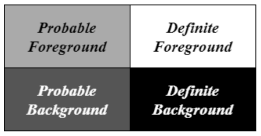

Figure 2. Color representationof the grabcut mask space values.

Pixels labelled as definite background or foreground remain unchanged after the application of the algorithm ; they’re said to be hard labelled.
It’s also worth to note that, by dropping the *Probable* vs *Definite* nuance, we can get back a regular binary mask at any time, as it is shown in steps 4-5 of Figure 3.

An initial mask needs to be provided which can range from being an approximation of the desired output, or a simple rectangular box containing the region of interest, as it is the case in steps 2-3 of Figure 3.

3.2. GrabCut outline

Iteratively, the algorithm will try to refine the mask, by taking into account the pixels color statistics as well as the

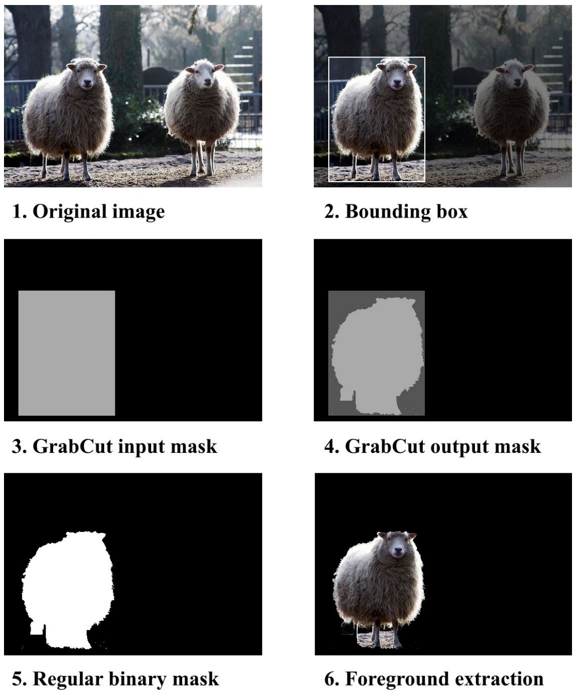

Figure 3. Full display of the GrabCut procedure with bounding box initialization.

foreground/background pixel distribution (given by the previous mask). The algorithm stops after convergence or alternatively, as it is the case in practice, after a finite number of iterations.

More in detail, GrabCut is built on the following loop.

1. Two Gaussian Mixture Model are used to model the foreground and background. Pixels are assigned a unique foreground/background GMM component according to their color statistics.

2. GMM parameters are learned.

3. Using the GMMs and the pixel color statistics, a weighted-graph is generated for the graph cut procedure [1]. Nodes represent pixels and neighbour pixels are connected. The graph is segmented into two regions, foreground and background, through a min cut whose cost function is the sum of the cut edges’ weights. The cost function is such that pixels within a region have similar colour and similar probabilities of being foreground/background, in order to maintain a certain object coherence.

4. A new mask is created, where non-hard labelled pixels are set to be probable foreground or probable background depending on the segmentation of their associated node in the weighted graph.

4. Methodology

GrabCut remains a foreground extraction algorithm. Unlike a CNN, it doesn't have any predictive capability. Left alone, GrabCut, may only extracts regions of an image blindly, without any awareness of the nature of the pixels being extracted. Nonetheless, the GrabCut output, more often than not, indeed corresponds to a coherent object, providing thus some premise for segmentation annotation.

Feeding a mask containing some inherent semantic information to GrabCut might allow us to achieve segmentation annotation.

CAM could embody such a mask. We're exploring this idea with the "GrabCut-CAM" technique, which is being described below.

4.1. GrabCut - CAM

Our GrabCut-CAM technique depends on 3 thresholds (t_0, t_1, t_2) hyper-parameters. Our algorithm can be described as follows:

Input: An image and a CAM

1. Transform w.r.t. (t_0, t_1, t_2) the CAM into a GrabCut mask.
2. Feed it to GrabCut.
3. **Return** GrabCut's output.

4.2. CNN reliant

To get our CAMs, we're making use of a slightly modified (following Section 2 ideas) convolutionnal neural network. This modified version is such that, when given a class and an image, the corresponding class activation map is returned, alongside the prediction.

4.3. CAM to GrabCut mask

The transformation (step 4.1.1) consists of *digitizing* (*i.e.* the process of mapping float intervals to integers) the CAM into a quaternary GrabCut mask. It's based on the assumption that high activation region (for class c) of an object (effectively of class c) should somewhat correspond to a foreground region, from GrabCut's point of view.

Let x, y be some spacial coordinates of an image. Consider an object of class c having a CAM M_c. We define $p = M_c(x, y)$, and assume that $p \in [0, 1]$. Our mapping then can be viewed as

$$p \longmapsto \begin{cases} \text{■ } \textit{Def. background} & 0 \le p < t_0 \\ \text{▩ } \textit{Pro. background} & t_0 \le p < t_1 \\ \text{▨ } \textit{Pro. foreground} & t_1 \le p < t_2 \\ \text{□ } \textit{Def. foreground} & t_2 \le p < 1 \end{cases}$$

where t_0, t_1, t_2 (with $0 \le t_0 < t_1 < t_2 \le 1$) are the thresholds hyper-parameters of our GrabCut-CAM technique.

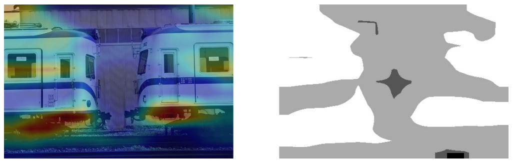

Figure 4. CAM to GrabCut mask.

Note that we may set the thresholds so that only probable GrabCut values may appear, as shown in Figure 5.

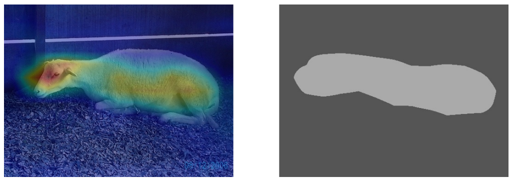

Figure 5. CAM to GrabCut mask with $t_0 = 0$, $t = 0.3$, $t_2 = 1$.

5. Experiment

5.1. Evaluated techniques

Three different version of our GrabCut-CAM methods are being tested. To ease their tuning, they only rely on one global hyper-parameter threshold t (with $0.1 \le t \le 0.9$). We have

1. GrabCut-CAM **PF PB** with $t_0, t_1, t_2 := 0, t, 1$ working with GrabCut masks only taking values in the $\{$*Probable foreground, Probable background*$\}$ set,
2. GrabCut-CAM **F PF** with $t_0, t_1, t_2 := 0, 0 + \delta, t$ working with GrabCut masks only [1] taking values in the $\{$*Definite foreground, Probable foreground*$\}$ set,
3. GrabCut-CAM **F PB** with $t_0, t_1, t_2 := 0, t, t + \delta$ working with GrabCut masks only taking values in the $\{$*Definite foreground, Probable background*$\}$ set.

where δ is an arbitrarily small parameter (set to 0.01) ensuring the strict monotonicity of the thresholds.

For comparison's sake two other techniques are also tested.

1. (naked) GrabCut and
2. "CAM" which is simply the transformation to the GrabCut mask (without calling GrabCut afterwards).

This will allow us to observe whether GrabCut can help CAMs or vice versa.

[1] Since the number of pixels being in range $[a, a + \delta]$ is negligible, we may say, by abuse, that the mask pixels may only take 2 values.

5.2. Evaluation

Our segmentation techniques are being tested on the PASCAL VOC - 2012 (training/validation) dataset, composed of 2913 images.
Both the object (with bounding boxes) segmentation and class segmentation tasks are being done.

GrabCut's ability is to distinguish between foreground and background *i.e.* is only able to do binary classification. From that arises an issue, as our GrabCut-CAM technique, which in the end gets its output from GrabCut, is supposed to perform pixel annotation (amongst multiple classes) which is an n-ary classification task.

More concretely, when using GrabCut sequentially (in order to extract the multiple classes/object regions), we might have redundancies (some pixel might be considered as foreground multiples times).
Therefore, GrabCut-CAM isn't really suited for strict pixel annotation (as one pixel could be predicted to belong to multiple classes, due to the GrabCut side effect).

To make up with these constraints, we can fall back on a (class wise) binary classification (instead of n-ary classification). As such, our segmentation methods will be aware of the class that their supposed to segment. Ideally, we'd wish that this evaluation process somewhat emulates strict semantic segmentation.

5.3. Implementation details

We're making use of OpenCV's GrabCut implementation with 1 iteration, and SqueezeNet [3], pre-trained on the ImageNet [2] dataset.

Since the dataset on which our CNN is trained isn't the same as the one on which it is tested, a translation between VOC and ImageNet classes had to be made. This translation relies on an ImageNet hierarchy [7] ; which maps meaningful ImageNet classes subset to more general classes, in a Cladogram fashion. As an example, the following ImageNet classes: *Chihuahua*, *Japanese spaniel* and *Maltese dog* can all be considered as *dog*.

5.4. Metrics

By taking into account Section 5.2, we will use the four standard binary classification measurements $\mathrm{TP}, \mathrm{FP}, \mathrm{FN}, \mathrm{TN}$. See Figure 6 to visualize them.
They're define as follows.

Consider an image whose region of class i must be extracted. Assume we're given the ground truth region and a prediction of the region to be extracted. We then have:

- TP_i the number of true positive for class i,
- FP_i the number of false positive for class i,
- TN_i the number of true negative for class i and
- FN_i the number of false negative for class i.

Two class wise metrics are being reported, namely, Intersection over Union (mIoU) and Accuracy (Acc). IoU is being extended to the whole dataset with mean IoU (mIoU) and frequency weighted IoU (fwIoU), whereas accuracy is extended to mean accuracy (mAcc).

Let N be the total number of VOC classes, i be some class and t_i be the total number of true pixels of class i having been seen in the whole dataset. We then formally define our metrics as:

$$\mathrm{IoU}_i = \frac{\mathrm{TP}_i}{\mathrm{TP}_i + \mathrm{FP}_i + \mathrm{FN}_i} \quad (4)$$

$$\mathrm{Acc}_i = \frac{\mathrm{TP}_i + \mathrm{TN}_i}{\mathrm{TP}_i + \mathrm{FP}_i + \mathrm{TN}_i + \mathrm{FN}_i} \quad (5)$$

$$\mathrm{mIoU} = \frac{1}{N}\sum_i \mathrm{IoU}_i \quad (6)$$

$$\mathrm{fwIoU} = \frac{1}{\sum_k t_k}\sum_i t_i\,\mathrm{IoU}_i \quad (7)$$

$$\mathrm{mAcc} = \frac{1}{N}\sum_i \mathrm{Acc}_i \quad (8)$$

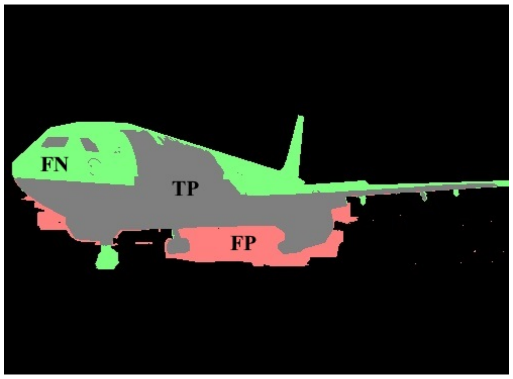

Figure 6. **I**ntersection **o**ver **U**nion visualization. The ground truth segmentation $(\mathrm{FN} + \mathrm{TP})$ is depicted in green whereas the prediction $(\mathrm{FP} + \mathrm{TP})$ is in red. The intersection (TP) is depicted in gray-khaki. The black region corresponds to the TN.

6. Results

6.1. Class segmentation

Technique \ Metric	mIoU	fwIoU	mAcc
cam 0.3	30.0	30.4	76.4
gccam PF PB 0.3	**34.9**	35.8	76.9
gccam F PF 0.7	20.2	22.4	31.6
gccam F PB 0.5	33.7	34.8	76.1

Table 1. Class segmentation measurements report

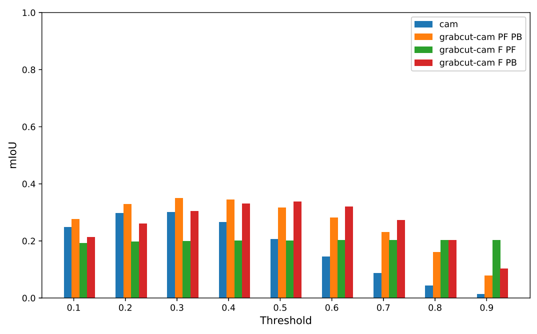

Figure 7. Class segmentation mIoU

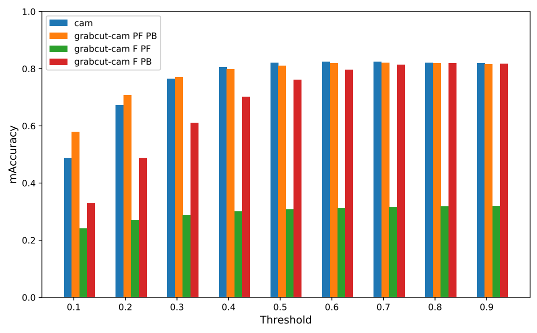

Figure 8. Class segmentation mAccuracy

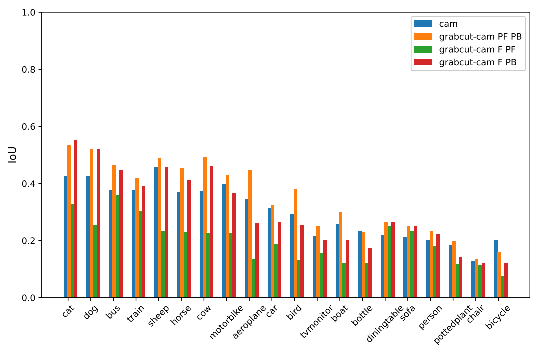

Figure 9. IoU by class for class segmentation with $t = 0.3$

6.2. Object segmentation

Technique \ Metric	mIoU	fwIoU	mAcc
grabcut	67.7	69.5	95.1
gccam PF PB 0.1	64.5	65.4	94.8
gccam F PF 0.6	**68.5**	70.1	95.0
gccam F PB 0.2	60.8	62.0	92.7

Table 2. Object segmentation measurements report

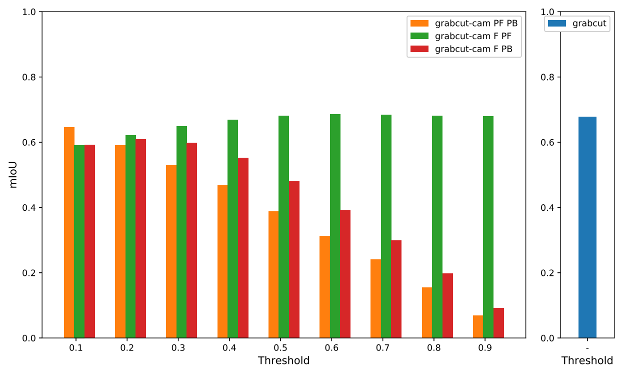

Figure 10. Object segmentation mIoU

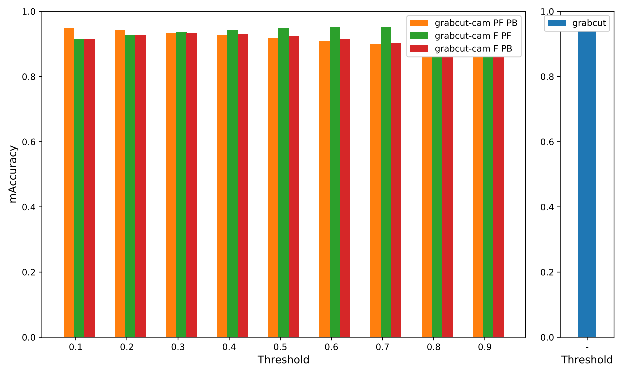

Figure 11. Object segmentation mAccuracy

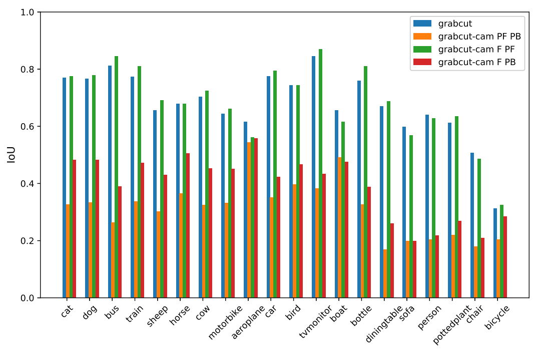

Figure 12. IoU by class for object segmentation with $t = 0.6$

For class and object segmentation respectively, the records of the 4 different techniques are displayed in the previous page.

- Tables Tab. 1 and Tab. 2 report mIoU, fwIoU and mAcc for the techniques (with their best respective threshold t).
- Figures Fig. 7 and Fig. 10 show the techniques mIoU for the different thresholds.
- Figures Fig. 8 and Fig. 11 show the techniques mAcc for the different thresholds.
- Figures Fig. 9 and Fig. 12 show the class IoUs of the different techniques with the given threshold t.

6.3. Result interpretation

One early observation, by refering to Table 1 and Table 2, is that GrabCut and CAM mutually help each other, as GrabCut-CAM has better result than both GrabCut and CAM, when taken apart.

The performance discrepancy between class and object segmentation is probably mainly due to the fact that object segmentation is done with bounding boxes, which constitutes a significant help. Moreover, objects are generally smaller and more *coherent* and have a more defined shape than a class region, which suits GrabCut better.

Except from GrabCut-CAM F PF, all techniques behave somewhat similarly w.r.t. to the threshold tuning ; having a convex plot, which peaks between 0.2 or 0.6.

7. Conclusion

References

[1] Yuri Y Boykov and M-P Jolly. Interactive graph cuts for optimal boundary & region segmentation of objects in nd images. In *Proceedings eighth IEEE international conference on computer vision. ICCV 2001*, volume 1, pages 105–112. IEEE, 2001. 2

[2] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009. 4

[3] Forrest N Iandola, Song Han, Matthew W Moskewicz, Khalid Ashraf, William J Dally, and Kurt Keutzer. Squeezenet: Alexnet-level accuracy with 50x fewer parameters and¡ 0.5 mb model size. *arXiv preprint arXiv:1602.07360*, 2016. 1, 4

[4] 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. 1

[5] Shervin Minaee, Yuri Y Boykov, Fatih Porikli, Antonio J Plaza, Nasser Kehtarnavaz, and Demetri Terzopoulos. Image segmentation using deep learning: A survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2021. 1

[6] Carsten Rother, Vladimir Kolmogorov, and Andrew Blake. “grabcut” interactive foreground extraction using iterated graph cuts. *ACM transactions on graphics (TOG)*, 23(3):309–314, 2004. 1, 2

[7] Mark Sandler, Andrew Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen. Mobilenetv2: Inverted residuals and linear bottlenecks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4510–4520, 2018. 4

[8] Bolei Zhou, Aditya Khosla, Agata Lapedriza, Aude Oliva, and Antonio Torralba. Object detectors emerge in deep scene cnns. *arXiv preprint arXiv:1412.6856*, 2014. 1

[9] Bolei Zhou, Aditya Khosla, Agata Lapedriza, Aude Oliva, and Antonio Torralba. Learning deep features for discriminative localization. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2921–2929, 2016. 1

[10] Yanzhao Zhou, Yi Zhu, Qixiang Ye, Qiang Qiu, and Jianbin Jiao. Weakly supervised instance segmentation using class peak response. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3791–3800, 2018. 1