QUALITY CONTROL IN CROWDSOURCED OBJECT SEGMENTATION

Anonymous

Author Affiliation(s)

ABSTRACT

This paper introduces several techniques to deal with noisy data in crowdsourced object segmentation tasks. We use the data collected with Click'n'Cut, an online interactive segmentation tool, and we perform several experiments towards improving the segmentation results. First, we introduce different superpixel-based techniques to filter users' traces, and assess their impact in the segmentation result. Second, we present different criteria to detect and discard the traces from potential bad users, resulting in a remarkable increase in performance. Finally, we show a novel superpixel-based segmentation algorithm which does not require any prior filtering and is based on weighting each user's contribution according to his/her level of expertise.

Index Terms— Object Segmentation, Crowdsourcing, Quality Control, Superpixel, Interactive Segmentation

1. INTRODUCTION

The problem of object segmentation is one of the most challenging ones in computer vision. It consists in, for a given object in an image, assigning to every pixel a binary value: 0 if the pixel is not part of the object, and 1 otherwise. Object segmentation has been extensively studied in various contexts, but still remains a challenge in general.

In this paper, we focus our experiments on interactive segmentation, that is, object segmentation assisted by human feedback. More specifically, we study the particular case in which the interactions come from a large number of users recruited through a crowdsourcing platform. Relying on humans to help object segmentation is a good idea since the limitations in the semantic interpretation of images is often the bottleneck for computer vision approaches.

In the crowdsourcing setup the users, also referred as *workers*, are not experts in the task they must perform and, in most cases, they face it for the first time. Workers tend to choose the task that can let them earn the most money in the minimum amount of time. From the employer perspective, crowdsourced online workers are more affordable than hiring task experts, and are also available in large numbers and a short recruiting time. However, many of these workers are also unreliable and do not meet the minimum quality standards required by the task. These situations motivate the need of a post-processing of the collected data to eliminate as many interaction as possible.

Quality control of workers' traces is a very active field of research, but is also widely dependent on the task. In computer vision, the quality of the traces is usually correlated with the visual content that motivated their generation. As an example, the left side of Figure 1 contain 3 points representing the labeling of three pixels: green points for foreground pixel and red points for the background ones. These same points may look coherent if assigned to different regions of support (middle) or inconsistent if providing contradictory labels for a given region (right). The creation of these regions through an automatic segmentation algorithm can assist in distinguishing between consistent or noisy labels.

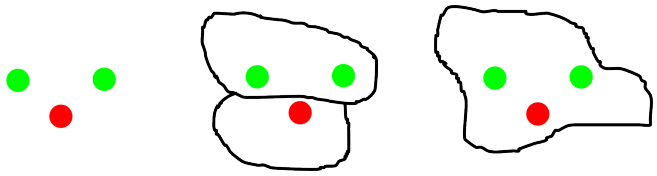

Fig. 1. The same set of foreground and background clicks (left) may be consistent (middle) or inconsistent (right) depending on the visual context.

This simple example illustrates the assumption that supports this work: computer vision can help filtering users' inputs as much as users' inputs can guide computer vision algorithms (towards better segmentation). Our contributions correspond to the exploration of three different venues for the filtering of human noisy interaction for object segmentation: filtering users, filtering clicks and weighting users' contributions according to a quality estimation.

This paper is structured as follows. Section 2 overviews previous work in interactive object segmentation and filtering of crowdsourced human traces. Section 3 describes the data acquisition procedure and Section 4 gives some preliminary results. Then, Section 5 introduces the filtering solutions and Section 6 explores a user weighted solution. Finally, Section 7 exposes the conclusions and future work.

This work has been developed in the framework of the project TEC2013-43935-R, financed by the Spanish Ministerio de Economía y Competitividad and the European Regional Development Fund (ERDF).

2. RELATED WORK

The combination of image processing and computer vision algorithms with human interaction has been extensively explored in the literature. Many works related to object segmentation have shown that user inputs throughout a series of weak annotations can be used to either seed segmentation algorithms or to directly produce accurate object segmentations. Researchers have introduced different ways for users to provide annotation for interactive segmentation. Some works [1, 2] ask users to draw the object's contour itself. Others seed segmentation algorithms with scribble based [3, 4] or click-based [5, 6, 7] annotations from users.

However, the performance of all these approaches directly relies on the quality of the traces that users produce, which raises the need of robust techniques for Quality Control of human traces.

The authors in [8] classify the three different sources of errors in these type of tasks, which are: "scammers", users who do not understand the task and users who just make random mistakes. They also point out the advantages of adding gold-standard images in the workflow to estimate the worker's accuracy.

Another common technique for quality control is to make users go through an in-depth tutorial before starting the actual task. Authors in [9] have demonstrated the need for tutorials by comparing the performance of trained and non trained users. In [2], users are discarded or accepted based on their performance in an initial training task and are periodically verified during the whole annotation process.

Quality control can also be a direct part of the experiment design. The Find-Fix-Verify design pattern for crowdsourcing experiments, originally developed in the context of spelling errors corrections, has been used in [10] for object detection. Some users draw bounding boxes around objects, others verify the quality of the bounding boxes, and a last category of users verifies whether all objects have been detected. Luis Von Ahn also formalized several methods for controlling quality of traces collected from Games With A Purpose (GWAP) [11].

A final control of the traces quality can be done at the end of the study. Authors of [12] explain how some task-specific observation help them discard users whose interaction patterns are unreliable. Ipeirotis et al [13] use the EM algorithm in order to estimate both a measure of worker's quality and of traces reliability, in a web page categorization task.

3. DATA ACQUISITION

The experiment is conducted using the interactive segmentation tool *Click'n'Cut* [5]. This tool allows users to label single pixels as foreground or background, and provides live feedback after each click by displaying the resulting segmentation mask overlaid on the image.

We used the data collected by [5] over two datasets:

- 96 images, associated to 100 segmentation tasks, are taken from the DCU dataset [4]. These images will be referred in the rest of the paper as our *test set*.
- 5 images are taken from the PASCAL VOC dataset [14]. These images form our *training set*.

Users were recruited on the crowdsourcing platform `Microworkers`. 20 users performed the entire set of 105 tasks, 4 females and 16 males, with ages ranging from 20 to 40 (average 25.6). Each worker was paid 4 USD when completing the 105 tasks.

4. CONTEXT AND EARLY RESULTS

The metric we use in this paper is the Jaccard Index, in order to be consistent and comparable to other state-of-the-art techniques. The Jaccard Index of two masks A and B can be computed as :

$$J(A,B) = \frac{A \cap B}{A \cup B} \tag{1}$$

A Jaccard of 1 is the best possible result (in that case $A = B$), and a Jaccard of 0 means that the two masks have no intersection.

On the test set, the top state-of-the-art methods have achieved an average Jaccard of 0.93. Authors of [5] claim that experts using *Click'n'Cut* reach an average Jaccard of 0.89. However, their paid workers perform significantly worse due to their high percentage of mistakes.

We have established that 35 % of the clicks that were produced by those 20 workers were incorrect. In [5], workers' are filtered based on their performance on the 5 training images. In the following sections, we present the different methods we use to improve the results.

5. DATA FILTERING

In this section we present three main approaches that focus on filtering the data that we have based on different criteria. First, we present several techniques to filter users' clicks based on their consistency with various superpixel algorithms. Second, we define and apply different rules to discard conflicting users. Finally, we study the possibility of combining both techniques.

5.1. Filtering clicks

In this subsection we work on the entire set of clicks at the same time. Whereas the results described in section 4 were averaged among users, here we consider the whole set of workers as one user. This is the actual definition of crowdsourcing, which consists in outsourcing the job ordinarily devoted to an expert to a crowd of workers.

Since there is a majority of correct clicks, we postulate that an incorrect click can be detected by looking at other clicks in its neighbourhood. Conflicts would then reveal potential incorrect clicks that should be filtered. The only problem with this idea is that, near an object boundary, clicks could be wrongly categorized as incorrect. In order to implement this idea, we would then need to consider neighborhoods both in the spatial and colorimetric spaces. This naturally leads to using superpixels.

We consider two popular superpixels algorithms : SLIC [15] and Felzenszwalb [16]. Once the image is oversegmented, there exist 6 different click distributions that can occur in a given superpixel (as shown in figure 2):

1. Higher number of foreground than background clicks
2. Higher number of background than foreground clicks
3. Same number of background and foreground clicks
4. Foreground clicks only
5. Background clicks only
6. No clicks

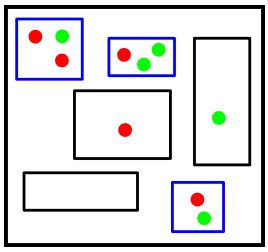

Fig. 2. Possible configurations of background (in red) and foreground (in green) clicks inside a superpixel. Conflicting superpixels are represented in blue.

Among these six configurations, the three first ones reveal conflicts between clicks. Two different methods can be considered to remove clicks and make the conflicts disappear:

- *Total removal of conflict clicks:* all the clicks in conflicting superpixels are removed.
- *Partial removal of conflict clicks:* only those clicks whose class is in minority in a superpixel are removed. In other words in configuration 1 (resp. configuration 2), background (resp. foreground) clicks are removed. In configuration 3, all clicks are removed.

These two methods are illustrated in figure 3.

Intuitively, total filtering removes a high number of clicks compared to partial filtering. The recall when finding errors must be higher with total filtering than with partial filtering, but the precision is lower.

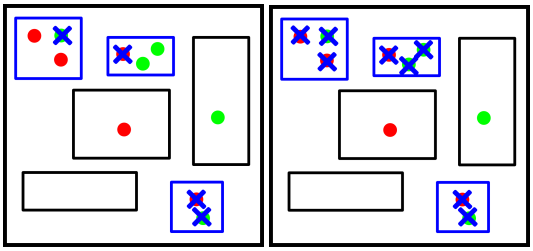

Fig. 3. Partial (on the left) and Total (on the right) filtering of the clicks introduced in figure 2.

	Partial Filtering	Total Filtering
SLIC [15]	0.21	0.24
Felzenszwalb [16]	0.21	0.22

Table 1. Jaccard Index obtained on the test set after applying partial and total filtering using [15] or [16].

Table 1 shows the Jaccard that we obtain on the test set when applying our two methods of partial and total filtering using two different superpixels techniques. The Jaccard without filtering is equal to 0.14, so the methods bring a significant improvement.

Since the results we achieve are not satisfying, we now explore an alternative way of filtering clicks: categorizing users.

5.2. Filtering users

In any crowdsourcing experiment having mistakes from users is the norm, not the exception. There are typically three main profiles of users who generate mistakes:

- users who understand the task, but are not focused or tired.
- spammers, who try to cheat the system to obtain the reward without effort.
- users who do not understand the task.

The two last classes refer to the most problematic users, who we would like to filter in priority. We use our training set as a Gold Standard to determine which users we should eliminate.

In order to separate good from bad users, we study two different features: users' error rate on the training set, and users' average Jaccard index on the training set.

We sort users based on those two features. Then for a given N, we select the N best users with respect to a feature and discard the remaining ones. Finally, we compute the average Jaccard of the selected users on the test set. The result of these steps is shown in figure 4.

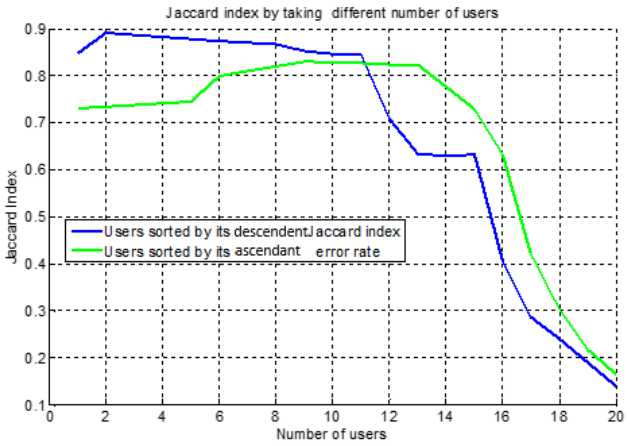

Fig. 4. Average Jaccard obtained when keeping an increasing number of users, with users sorted according to their performance on the training set. Sorting users according to Jaccard is shown in blue, and according to their error rate in green.

It is interesting to note that the blue curve is almost perfectly decreasing, which suggests a high correlation beween the average Jaccard on the training set and the average Jaccard on the test set. On the other hand, the green curve is bell-shaped which suggests that the best-performing users (in terms of Jaccard Index) are not those who make the least errors.

There are several conclusions that can be drawn from this curve. First, there is no obvious correlation between error rate and performance (i.e. Jaccard Index for the problem of object segmentation). Indeed, a user who would produce only one correct click per image would have a null error rate but a very low jaccard. In fact, the error rate is not discriminant enough to filter out users since spammers do not necessarily make a lot of mistakes, users who do not understand the task may still produce valid clicks, and good users may get tired and produce a lot of errors on a few images. It seems much more effective to filter users based on their actual performance on the task, since at least users who did not understand the task would make a very poor performance.

Looking at the blue curve in figure 4, we see that the best Jaccard is achieved if we only keep two users and goes up to 0.9 which is comparable to what expert users have reached (see section 4). We could argue that two users is not a significant enough number and that reaching such a high value could be a statistical anomaly, but even if we keep half the users the average Jaccard obtained on the test set is nearly 0.85 which is already a very good result.

At this point, we have shown that filtering users dramatically increases the overall performance. We will study in the next subsection if we can add an additional step to this operation.

5.3. Filtering clicks and users

We now study whether, once users have been filtered based on their average Jaccard on the training set, we can use the techniques presented in section 5.1 to further denoise the remaining set of clicks. The intuition is that after filtering the users, there should remain a large majority of correct clicks and that the filtering techniques of section 5.1 are more efficient if the overall error rate is low.

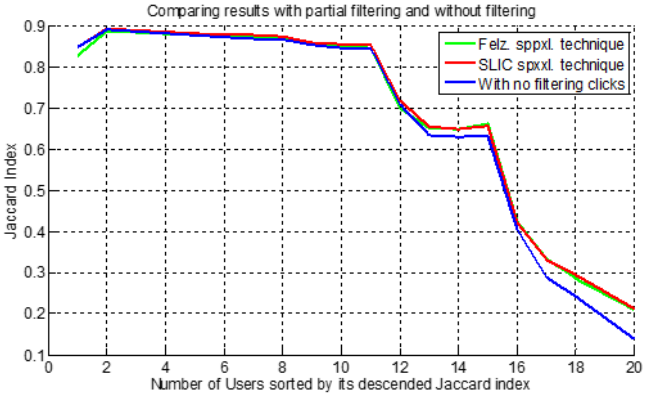

Fig. 5. Average Jaccard obtained when keeping an increasing number of users, with users sorted according to their Jaccard on the training set. The blue curve is without additionnal filtering, the red curve is with a partial filtering using SLIC [15] and the green curve is with a partial filtering using Felzenszwalb [16].

Figure 5 shows the Jaccard curves that we obtain when applying partial filtering after user filtering. We can see that there is no major effect in filtering clicks when we keep a low number of users, but that the effect is more significant as we keep more and more users.

Figure 6 shows the Jaccard curves that we obtain when applying total filtering after user filtering. In that case, filtering clicks provokes a severe drop in performance when we keep only a few users, and has mostly the same effect as partial filtering otherwise. This is probably explained by the fact that we remove too many clicks in the total filtering technique.

6. DATA WEIGHTING

In the previous section, we have seen how removing some of the data could improve the segmentation results. The main drawback of this approach is that among the clicks we are discarding, some are valid and bring useful information that could be used to obtain a better segmentation. This is why in this section we focus on designing an algorithm that uses the entire set of clicks without filtering.

Instead of filtering the clicks, we simply leverage their contribution based on an analysis of the users. For example,

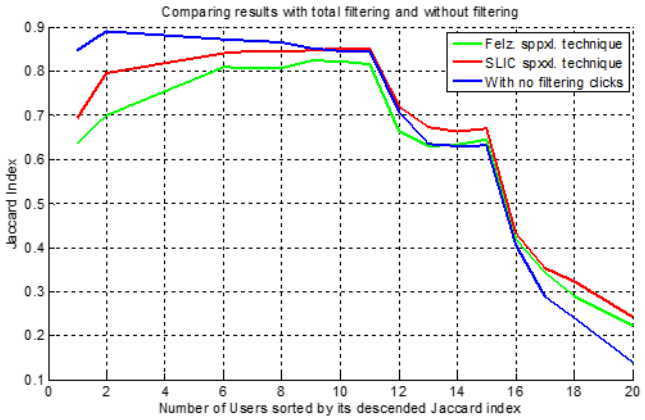

Fig. 6. Average Jaccard obtained when keeping an increasing number of users, with users sorted according to their Jaccard on the training set. The blue curve is without additional filtering, the red curve is with a total filtering using SLIC [15] and the green curve is with a total filtering using Felzenszwalb [16].

if a worker i has a 5% error rate on gold standard images, the measure of confidence c_i for this worker will be 0.95.

Then we consider a superpixel segmentation and apply the following algorithm to "paint" the superpixels.

1. Initialize the weight of each superpixel to 0
2. For each click,
   (a) get the measure of confidence c of the user who clicked it. The contribution of this click will be c (resp. $1 - c$) if it is labeled foreground (resp. background).
   (b) add the click contribution to the superpixel it falls into
3. For each superpixel, divide the final weight by the number of clicks that fell into it. If there was no click, the weight of the superpixel is 0.

At this step, the painted superpixels form a *foreground map* of the image, with values ranging from 0 (maximum confidence of *background*) to 1 (maximum confidence of *foreground*).

Intuitively, we give more confidence to a click when there are few superpixels on the segmentation, and less confidence when there are a lot of superpixels (because the surface that is painted is smaller). So, if we use an oversegmentation with few superpixels we take the risk of putting too much confidence into incorrect clicks, whereas an oversegmentation with many superpixels will limit the influence of incorrect clicks (but also from correct clicks). In order to reduce this dependency to the number of superpixels, we use several oversegmentations with different parameters and produce a foreground map for each one of them. Finally, the whole set of foreground maps (one for each over-segmentation) is pooled by averaging their normalized pixel values. This late averaging of weighted superpixels is the main protection against errors in collected clicks.

The set of oversegmentations is obtained by running the segmentation algorithm proposed by Felzenswalb [16] with a range of values of its parameter k (10, 20, 50, 100, 200, 300, 400, 500). In addition we use partitions generated by the SLIC superpixel algorithm [15], with a range of values of the parameter *initial region size* (5, 10 20, 30, 40, 50). We have established that this combination was the one who gave the better results on the training set.

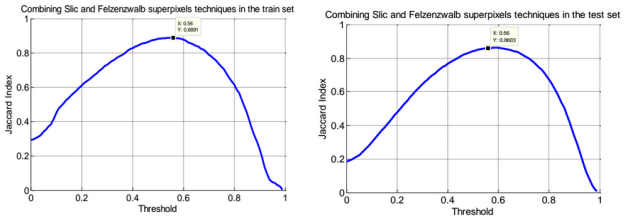

Fig. 7. Average Jaccard obtained from different thresholding of the Foreground map on the training set (left) and on the test set (right)

Then, we show on figure 7 how to establish a threshold to obtain a binary mask out of the Foreground map. We learn from our training set that the best segmentation results are obtained with a threshold equal to 0.56. Applying this threshold on our test set, we obtain a final Jaccard index equal to 0.86.

Figure 8 gives two examples of the obtained foreground maps. The object to be segmented is the brightest region, and traces from noisy clicks can be seen where regions in the background are bright as well.

Fig. 8. Foreground map of object segmentation based on weighted worker's clicks.

7. CONCLUSION

8. REFERENCES

[1] Bryan C Russell, Antonio Torralba, Kevin P Murphy, and William T Freeman, "Labelme: a database and web-based tool for image annotation," *International journal of computer vision*, vol. 77, no. 1-3, pp. 157–173, 2008.

[2] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C. Lawrence Zitnick, "Microsoft coco: Common objects in context," *CoRR*, 2014.

[3] Carsten Rother, Vladimir Kolmogorov, and Andrew Blake, "Grabcut: Interactive foreground extraction using iterated graph cuts," in *ACM Transactions on Graphics (TOG)*. ACM, 2004, vol. 23, pp. 309–314.

[4] Kevin McGuinness and Noel E. O'Connor, "A comparative evaluation of interactive segmentation algorithms," *Pattern Recognition*, vol. 43, no. 2, 2010.

[5] Axel Carlier, Vincent Charvillat, Amaia Salvador, Xavier Giro-i Nieto, and Oge Marques, "Click'n'cut: crowdsourced interactive segmentation with object candidates," in *Proceedings of the 2014 International ACM Workshop on Crowdsourcing for Multimedia*. ACM, 2014, pp. 53–56.

[6] Amaia Salvador, Axel Carlier, Xavier Giro-i Nieto, Oge Marques, and Vincent Charvillat, "Crowdsourced object segmentation with a game," in *Proceedings of the 2nd ACM international workshop on Crowdsourcing for multimedia*. ACM, 2013, pp. 15–20.

[7] Pablo Arbeláez and Laurent Cohen, "Constrained image segmentation from hierarchical boundaries," in *Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on*. IEEE, 2008, pp. 1–8.

[8] David Oleson, Alexander Sorokin, Greg P Laughlin, Vaughn Hester, John Le, and Lukas Biewald, "Programmatic gold: Targeted and scalable quality assurance in crowdsourcing.," *Human computation*, vol. 11, pp. 11, 2011.

[9] Luke Gottlieb, Jaeyoung Choi, Pascal Kelm, Thomas Sikora, and Gerald Friedland, "Pushing the limits of mechanical turk: qualifying the crowd for video geo-location," in *Proceedings of the ACM multimedia 2012 workshop on Crowdsourcing for multimedia*. ACM, 2012, pp. 23–28.

[10] Hao Su, Jia Deng, and Li Fei-Fei, "Crowdsourcing annotations for visual object detection," in *Workshops at the Twenty-Sixth AAAI Conference on Artificial Intelligence*, 2012.

[11] Luis Von Ahn and Laura Dabbish, "Designing games with a purpose," *Communications of the ACM*, vol. 51, no. 8, pp. 58–67, 2008.

[12] Andrew Mao, Ece Kamar, Yiling Chen, Eric Horvitz, Megan E Schwamb, Chris J Lintott, and Arfon M Smith, "Volunteering versus work for pay: Incentives and tradeoffs in crowdsourcing," in *First AAAI Conference on Human Computation and Crowdsourcing*, 2013.

[13] Panagiotis G Ipeirotis, Foster Provost, and Jing Wang, "Quality management on amazon mechanical turk," in *Proceedings of the ACM SIGKDD workshop on human computation*. ACM, 2010, pp. 64–67.

[14] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman, "The pascal visual object classes (voc) challenge," *IJCV*, vol. 88, no. 2, 2010.

[15] Radhakrishna Achanta, Appu Shaji, Kevin Smith, Aurelien Lucchi, Pascal Fua, and Sabine Susstrunk, "Slic superpixels compared to state-of-the-art superpixel methods," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 34, no. 11, pp. 2274–2282, 2012.

[16] Pedro F Felzenszwalb and Daniel P Huttenlocher, "Efficient graph-based image segmentation," *IJCV*, vol. 59, no. 2, 2004.