Automatic Salient Object Extraction with Contextual Cue

Le Wang, Jianru Xue, Nanning Zheng
Institute of Artificial Intelligence and Robotics
Xi'an Jiaotong University, China
{lwang,jrxue,nnzheng}@aiar.xjtu.edu.cn

Gang Hua
Department of Computer Science
Stevens Institute of Technology, USA
ganghua@gmail.com

Abstract

We present a method for automatically extracting salient object from a single image, which is cast in an energy minimization framework. Unlike most previous methods that only leverage appearance cues, we employ an auto-context cue as a complementary data term. Benefitting from a generic saliency model for bootstrapping, the segmentation of the salient object and the learning of the auto-context model are iteratively performed without any user intervention. Upon convergence, we obtain not only a clear separation of the salient object, but also an auto-context classifier which can be used to recognize the same type of object in other images. Our experiments on four benchmarks demonstrated the efficacy of the added contextual cue. It is shown that our method compares favorably with the state-of-the-art, some of which even embraced user interactions.

1. Introduction

The problem of extracting a foreground object from a single image has wide applications in both computer vision and computer graphics [12, 16]. The vast majority of recent research on this topic had adopted an energy minimization formulation [5, 16, 4], which incorporates a *data term* that models the appearances of the foreground and background, and a *spatial prior term* that is often intended to re-enforce the smoothness of the labels.

Typical energy formulation adopts either the Markov Random Fields (MRFs) [12] or Conditional Random Fields (CRFs) [13], which is subsequently optimized by energy minimization algorithm such as belief propagation [25] and its variants [22], or graph-cut [5, 4]. Nevertheless, most previous energy formulations for image segmentation only model the appearance cues such as color or texture in their data term, which neglected valuable high level information such as visual context.

Context comes with a variety of forms, and can be referred to as Gestalt laws in middle level knowledge regarding intra-object configurations and inter-object relationships [3, 19]. There are many methods employing context cues for object categorization [15, 17] and multi-class segmentation [9], but few methods using context aimed at foreground object segmentation are proposed. Intuitively, they should provide beneficial information for separating a foreground object from its background. This motivated us to explore the usage of context information for the task of automatic salient object extraction from a single image.

In particular, we cast the auto-context model by Tu [20] into an energy minimization formulation to improve both the efficiency and accuracy for foreground object segmentation. The auto-context model builds a multi-layer Boosting classifier on image features and context features surrounding a pixel to predict if this pixel is associated with the target concept, where subsequent layer is working on the classification maps from the previous layer. Hence through the layered learning process, it automatically takes more spatial context into consideration when classifying one pixel.

Nevertheless, learning both the appearance model and the auto-context model of the foreground/background necessitates a set of labeled image pixels with both negative and positive examples. Some previous approaches have either resorted to user interactions to provide such labels, such as GrabCut [16] and Lazy Snapping [12], or made assumptions on the location of the object of interest [11]. Notwithstanding the efficacy of user interaction, it is still desirable to have a fully automated system to extract the salient object from a single image.

The notion of a "salient" object could have multiple implications. In this paper, we regard a visual object to be salient if it accounts for a significant portion of the image. To achieve a fully automated system, we resort to a graph-based computational attention model [10] to bootstrap our energy minimization process. We employ the saliency map to generate an initial segmentation, which is subsequently used to train the appearance model and the first-layer Boosting classifier of the auto-context model. Once these models are obtained, we perform graph-cut (i.e., min-cut/max-flow) [5, 4] to produce a new segmentation.

The new segmentation then serves to re-estimate the ap-

pearance model and the subsequent layer of Boosting classifier of the auto-context model. This process iterates until convergence, which returns not only an automatic segmentation of the salient object, but also a fully trained auto-context model. This automatically learned context model can indeed be applied to recognize the same type of object in new images.

In our formulation, we also utilize the visual saliency cue to compute the the weights for fusing a set of low-level features to form our appearance model, where the weight of each feature is in proportion to its contribution to the saliency map. This feature fusion process is shown to produce more robust appearance model of the foreground object. In summary, the key contributions of this paper are on the following aspects:

- An automatic segmentation method that can perform segmentation of the object of interest from its background without any user intervention.
- A unified energy minimization formulation which leverages saliency cue, appearance cue and contextual cue in the data term.
- An iterative algorithm to jointly estimate the segmentation of the foreground object, and learn the auto-context model which can be used to recognize the same type of object in new images.

To our best knowledge, we are the first to have explored contextual and saliency cues in an energy minimization framework for automatic object of interest extraction.

2. Energy Formulation with Contextual Cue

We cast an energy minimization formulation for salient object extraction. Given an image $\mathbf{I}$, each pixel $p \in \mathbf{I}$ will be assigned a binary label $L_p \in \{0, 1\}$, where $0/1$ corresponds to the background/foregroud, respectively. Our objective is to identify a labeling L that minimizes $E(L)$, i.e.,

$$E(L) = \lambda \sum_{p \in \mathbf{I}} (C_p(L_p) + D_p(L_p)) + \sum_{(p,q) \in N} \omega_{pq} \delta(L_p, L_q), \tag{1}$$

where $C_p(L_p)$ and $D_p(L_p)$ measure the cost of labeling pixel p to be L_p from an auto-context model and an appearance model, respectively. The sum of $C_p(L_p)$ and $D_p(L_p)$ composes the *data term* in our model. The function $\delta(L_p, L_q)$ is a Dirac delta function. In our implementation, ω_{pq} is computed based on the edge probability map from the Berkeley boundary detection system [14], which incorporates texture, luminance and color cues. The difference is that they [14] measure the dissimilarity of adjacent pixels, while ours measures the similarity between adjacent pixels p and q. Naturally, $\omega_{pq}\delta(L_p, L_q)$ composes the *spatial prior term* to encourage piece-wise smooth labeling L. The coefficient $\lambda \geq 0$ controls the relative weight of the data term and the spatial prior term.

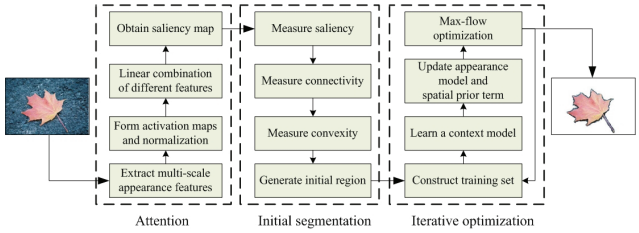

Figure 1. The flow chart of our automatic system.

The particular contextual model we adopted is the auto-context model proposed by Tu [20], which seeks for a multi-layer Boosting classifier, with subsequent Boosting classifier working on the classification maps of the previous layer. Hence, $C_p(L_p)$ relies on the discriminative probability (or classification) maps created by a learned auto-context classifier. If we know the auto-context model $C_p(L_p)$ and the appearance model $D_p(L_p)$, the energy function $E(L)$ can be efficiently solved by leveraging graph-cut [5, 4] to obtain an optimal solution. However, these models often need to be pre-trained with a set of labeled training examples, which is not always available. This is why many previous methods even resort to user interactions to obtain the model.

Our target is a fully automatic system, so we need to jointly estimate the segmentation, the auto-context model and the appearance model. Intuitively, an iterative optimization process is needed. We initialize such an iterative algorithm by obtaining the very initial segmentation by using a bottom-up visual saliency model [10].

We proceed to present our iterative algorithm in Section 3, the auto-context model $C_p(L_p)$ in Section 4, and the appearance model $D_p(L_p)$ in Section 5, which is fused from a number of low-level features [24].

3. Iterative Optimization

Bootstrapped by a visual saliency model [10], in each iteration of our algorithm, we first update the auto-context model, the appearance model and the spatial prior term, and then minimize the energy via graph-cut [5, 4]. This process iterates until it converges, as illustrated in Fig.1.

3.1. Generate the Initial Region

We use a bottom-up visual saliency model, namely the GBVS [10] to locate and create the most salient region of the image. The saliency map is generated by combining multi-scale image features including color, intensity and orientation into a single topographical saliency map [23]. Since the most salient region is often associated with the most salient object, we select it as the initial region of the salient object.

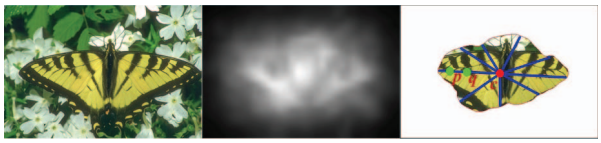

Figure 2. Left: input image. Middle: saliency map generated by a generic saliency model. Right: An example of the convexity measure calculated on the initial region of the salient object. First, the center of the initial region of the salient object c marked with a red dot is selected; then the straight lines passing through c are shot out from c; finally, for any point p marked with green dot on the line inside the object, we just assess whether any point q also marked with green dot on the straight line connecting c and q is also inside the object. If so, the convexity is satisfied.

Obviously, a good initial segmentation is necessitated to obtain a good final segmentation result. Then the question becomes: *how can we measure the quality of an initial segmentation?* We identified three useful measures: (1) **Connectivity**, which requires the initial region to be a single region with closed contour. 2) **Convexity**, which requires the contour of the region to be convex. We adopt the algorithm in [21] to compute the "convex" measure, as illustrated in Fig. 2. 3) **Saliency**, which indicates that the most salient region shall be more likely to be a good initial segmentation.

We design an adaptive selection mechanism to select the minimal connected region of the saliency map as the initial segmentation of the salient object to satisfy these three measures. Specifically, the region in the saliency map with equal or greater saliency than a threshold (e.g., 90% of the maximal saliency value of the saliency map) is firstly selected as a candidate region. This will first meet the "saliency" measure. Secondly, if the candidate region satisfies the "connectivity", and to some extent meet the "convexity" measure, we then use it as the initial region of the salient object. Otherwise the candidate region is discarded, and we reduce the threshold by 5% of the maximal saliency. The aforementioned procedure is repeated until a minimal region is found. Fig. 2 presented an illustrative example.

3.2. Iterative Process

By treating the identified salient region as the foreground, and the rest of the image as the background, the iterative process is started and performed until convergence. In each iteration, the auto-context model, the appearance model and the spatial prior term of the energy in Eq(1) are updated separately based on the segmentation and the discriminative probability maps of the previous iteration, where the discriminative probability maps are estimated by the auto-context model. To have a clear understanding of the algorithm, we defer the details of how to update the auto-context and appearance models to Section 4 and Section 5, respectively. The updated energy is then minimized by performing a max-flow algorithm [6] twice to take both the shrinkage and expansion of the salient object into account. This is a heuristic we identified and empirically it helps to converge faster.

The first max-flow algorithm is performed within the foreground region, which means that only pixels belonging to foreground are allowed to flip their labels. A new segmentation map S_f and the corresponding energy E_f are obtained. The second max-flow algorithm is performed in the background, only background pixels are allowed to flip their labels. A new segmentation S_b and the corresponding energy E_b are also obtained. We then choose the one with lower energy from these two segmentations to be the output of current iteration. This process iterates until convergence and we take the final segmentation as our ultimate result. Fig.3 illustrates the iterative process of segmentation.

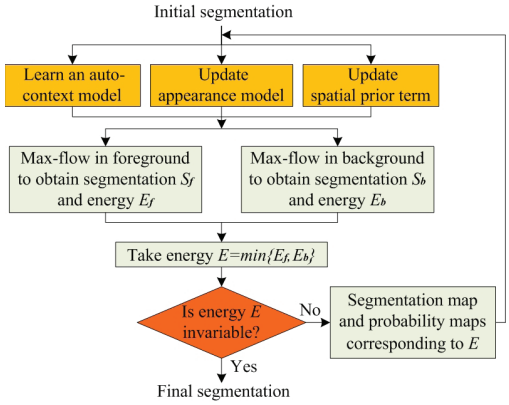

Figure 3. The iterative process of the energy minimization.

4. Auto-context Model

Contextual dependency is one major visual cue in segmentation, which has not been well explored by previous energy minimization formulation. To better determine how fit a pixel belongs to foreground or background by including a large amount of contextual information, we resort to learn an auto-context model iteratively (For details about the auto-context model itself, please refer to Tu [20]), adapting each iteration of the energy minimization in Eq(1). To achieve this in the iterative learning process, we firstly define an adaptive sampling strategy to collect samples for training the auto-context model, and subsequently use it to update the model.

4.1. Sampling Structure

As illustrated in Fig. 4, with a given segmentation map and the discriminative probability maps from the previous iteration, a training set for the auto-context model is formed

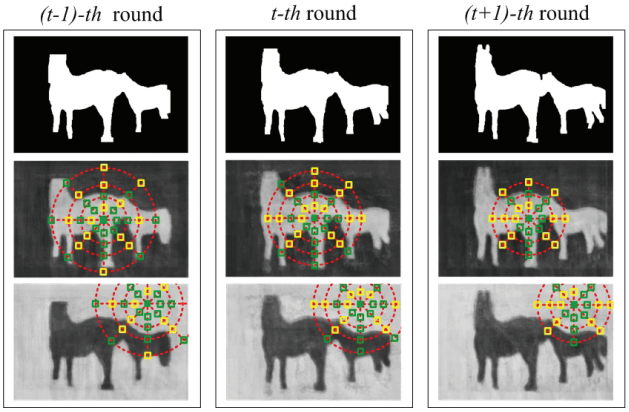

Figure 4. Collecting training samples using the multi-scale sampling structure on consecutive three rounds of the iterative learning process of the auto-context model. The first row are segmentation maps, the 2nd and 3rd rows are discriminative probability maps. The patches in yellow are uniformly sampled along the circles. The patches in green are selected context features to construct the strong classifier.

by including a large amount of patches centered at each pixel location of the discriminative probability maps along with its label. It consists of two sets of patches: a set of positive patches which are centered around foreground pixel locations, and a set of negative patches centered around background pixel locations (which are those locations far from the foreground region).

Instead of using all pixels around a pixel location of interest to extract the patches, we define a multi-scale sampling structure for each pixel location, and sample patches along the circles of the structure to form the training set. This is different from the original auto-context algorithm [20] in which the model is learned from a class of labeled images. In our design, for each pixel location, the sampling structure of patches first includes the pixel locations within 3 pixels away from the current pixel location; and further, circles centered at the current pixel location with different radiuses are built, and patches are sparsely sampled along these circles in 45° intervals, as shown in Fig. 4.

4.2. Update the Auto-context Model

In the first round of the iterative learning process, the training patches set for the auto-context model is built as

$$S_1 = \{(L_p, P(N_p)), p = 1, \ldots, n\},$$

where n is the number of pixel samples, $P(N_p)$ denotes the local image patch centered at pixel p (we use local image patches of fixed size 11×11). The haar features are then extracted from this patch to form the appearance feature vector of the current pixel p.

After the first classifier is learned on the appearance feature vector extracted from the local image patch $P(N_p)$, the discriminative probabilities $\mathbf{p}_p^{(1)}$ for each pixel p on the discriminative probability maps output by the learned classifier are used as contextual cue (individual probabilities or the mean probability within a 3×3 patch). The auto-context model $C_p(L_p)$ in Eq(1) for pixel p is then updated as

$$\begin{aligned} C_p^{(1)}(L_p) = \mathbf{p}_p^{(1)} &= p(L_p|P(N_p)), \\ \sum_{L_p} C_p^{(1)}(L_p) &= 1, \\ \forall p \in \mathbf{I}. & \end{aligned} \tag{2}$$

From the second round of the iterative learning process, we construct the training patches set as

$$S_2 = \{(L_p, (P(N_p), O^{(1)}(p))), p = 1, \ldots, n\},$$

where $O^{(1)}(p)$ are patches on the sampling structure centered at pixel p, which is sampled from the discriminative probability maps $\mathbf{P}^{(1)} = \{\mathbf{p}_p^{(1)} | p \in \mathbf{I}\}$ of the previous round. $P(N_p)$ is the local image patch centered at pixel p as before. As discussed above, $O^{(1)}(p)$ is a collection of patches of the sampling structure sampled on the discriminative probability maps obtained from previous round, and hence can be expressed as

$$O^{(1)}(p) = \{O(p, r_i, \theta_j), i = 1 \ldots N_r, j = 1 \ldots N_\theta\},$$

where r_i and θ_j denote the radius and angle, i and j are the indices of radiuses and angles, N_r and N_θ are the total numbers of radiuses and angles, respectively. $O(p, r_i, \theta_j)$ is the patch with radius r_i and angle θ_j away from the pixel p. More specifically, $r_i \in \{1, 3, 5, 7, 10, 12, 15, 20, 25, 30, 35, 40, 45, 50, 60, 70, 80, 90, 100, 125, 150, 175, 200\}$(radius sequence). N_r is initialized to be 23 (the number of the radius sequence) and is gradually reduced and remains unchanged until it reaches a minimum (4 or 5), and N_θ is fixed to be 8 in 45° intervals.

Then, a new classifier is trained on the probabilities of patches of the sampling structure $O^{(1)}(p)$ sampled on the discriminative probability maps, and on the appearance features extracted from the local image patch $P(N_p)$. The auto-context model is then updated as follows

$$\begin{aligned} C_p^{(2)}(L_p) = \mathbf{p}_p^{(2)} &= p(L_p|P(N_p), O^{(1)}(p)), \\ \sum_{L_p} C_p^{(2)}(L_p) &= 1, \\ \forall p \in \mathbf{I}, & \end{aligned} \tag{3}$$

where $\mathbf{p}_p^{(2)}$ denotes the discriminative probabilities on the new discriminative probability maps $\mathbf{P}^{(2)} = \{\mathbf{p}_p^{(2)} | p \in \mathbf{I}\}$ created by the new learned classifier.

This process will iterate until it converges where the discriminative probability maps are not changing anymore. Table 1 presented a summarization on one round of the iterative process of learning the auto-context model. Indeed, in our formulation, the auto-context model is iteratively updated seamlessly with the iterative minimization of the energy in Eq(1).

Input: the input image, the segmentation map and the discriminative probability maps from previous iteration.
Output: the learned auto-context model at current *t-th* iteration

- Construct a training set

$$S_t = \{(L_p, (P(N_p), O^{(t-1)}(p))), p = 1, \ldots, n\}$$

- Train a classifier on both image appearance and context features extracted from $P(N_p)$ and $O^{(t-1)}(p)$ respectively.
- Use the trained classifier to compute new discriminative probability maps $\mathbf{P}^{(t)} = \{\mathbf{p}_p^{(t)} | p \in \mathbf{I}\}$.

The output is exactly the auto-context model presented in Eq(1) $C_p(L_p) = \mathbf{p}_p^{(t)} = p(L_p|P(N_p), O^{(t-1)}(p))$

Table 1. Example of one round of the iterative process of learning the auto-context model.

It should be noted that the sampling structure of the auto-context model on the discriminative probability maps is in a contractive fashion, i.e., at the beginning of the iteration, the sampling structure is large to cover the boundary of the object and then it reduces as the segmentation map is better and better. At the beginning, large sample steps are taken to generate the patches due to less accuracy of the segmentation map. With the iterations, the sampling step is reduced to find fine-gained boundary of the foreground object. Fig. 4 clearly illustrated this point on consecutive three rounds of the iterative learning process.

5. Appearance Model

Our appearance model $D_p(L_p)$ in Eq (1) fuses color and intensity as

$$D_p(L_p) = \omega_i D_p^i(L_p) + \omega_c D_p^c(L_p), \tag{4}$$

where $D_p^i(L_p)$ and $D_p^c(L_p)$ are simply the intensity distribution and color distribution, while ω_i and ω_c specify the importance of intensity cue and color cue in composing the appearance model $D_p(L_p)$, respectively.

The appearance model is also updated in each iteration of the energy minimization. As a segmentation obtained from the previous iteration, intensity distribution $Pr(I_p|''Fg'')$ for the foreground and $Pr(I_p|''Bg'')$ for the background can be approximated by histograms of intensities of pixels belonging to the foreground and background, respectively. These histograms are then used to calculate $D_p^i(L_p)$, i.e.,

$$\begin{aligned} D_p^i(L_p = 1) &= -ln(Pr(I_p|''Fg'')), \\ D_p^i(L_p = 0) &= -ln(Pr(I_p|''Bg'')), \end{aligned} \tag{5}$$

where I_p is the intensity of pixel p. Similarly, the term $D_p^c(L_p)$ is obtained as

$$\begin{aligned} D_p^c(L_p = 1) &= -ln(Pr(C_p|''Fg'')), \\ D_p^c(L_p = 0) &= -ln(Pr(C_p|''Bg'')), \end{aligned} \tag{6}$$

where C_p is the RGB color vector of pixel p.

It is worth noting that, the weights ω_i and ω_c are adaptively selected based on the visual saliency model. First, color and intensity features are normalized to maintain the same dimension. Then the values of the weights ω_i and ω_c are assigned in proportion to their respective contributions to the saliency map generated by the bottom-up visual saliency model [10].

6. Experiments and Discussion

Test database. In order to evaluate the performance of our method, we test it on 4 image segmentation benchmarks including the Berkeley segmentation database [14], the GrabCut database[16], the Weizmann segmentation database[1] and the MSRC database[19].

Experiment 1. Performance of our method. Fig. 5 shows some sample segmentation results from the above mentioned 4 databases using our proposed method. Empirical results show that our method is able to extract important part of some of the difficult salient objects, and is able to deal with weak boundaries, and complex object and background without any user intervention. We highly recommend to check out our supplementary demo video for more results on *www.aiar.xjtu.edu.cn/personal*.

To be noted that, λ in Eq(1) is fixed to be 5 throughout our empirical evaluation. In addition, due to the time consuming in learning the auto-context model, the running time on a computer of 2.99 GHz CPU and 2.0 GB RAM are several minutes per image with our Matlab implementation.

Experiment 2. Evaluation of the auto-context model. To evaluate the importance and usefulness of the auto-context model in the segmentation, we implemented two versions of our method. Their only difference is that one includes the auto-context model in the energy function, and the other does not include it. Experimental results in Fig. 6 clearly demonstrate that the auto-context model is important and very effective. For example, for the horse image in Fig. 6, the success of cutting out the legs of the horses is obviously due to the auto-context model.

Experiment 3. Convergence analysis. According to the experimental results, each step of our energy minimization ensures that the energy in Eq(1) is non-increasing, so it can

Figure 5. Segmentation results of our method. The rows 1, 3, 5, 7 and 9 are some test images from 4 segmentation benchmarks including the Berkeley[14], GrabCut[16], Weizmann[1] and MSRC database [19]. The rows 2, 4, 6, 8 and 10 are the corresponding segmentation results, respectively.

always converge. Fig. 7 illustrates the trend of energy function tested on the 3 images in Fig. 6. The curve shows that the energy function always converges within 30 iterations.

Experiment 4. Objective evaluation. We also compare our method with 5 state-of-art algorithms listed in Table 2 by evaluating them on the Weizmann single object segmentation database [1], which contains 100 images along with ground truth segmentations. The segmentation is evaluated by assessing its consistency with the ground truth segmentation. The average F-measure score on the entire database is computed to serve as the final score. Indeed, the F-measure is the harmonic mean of precision and recall measures calculated on the foreground pixels, i.e.,

$$F = \frac{2 \cdot Recall \cdot Precision}{Recall + Precision}. \quad (7)$$

Table 2 summarized the F-measures on the single object database for all the competing methods, the scores of the competing methods are directly quoted from *www.wisdom.weizmann.ac.il/˜vision/Seg_Evaluation_DB/*

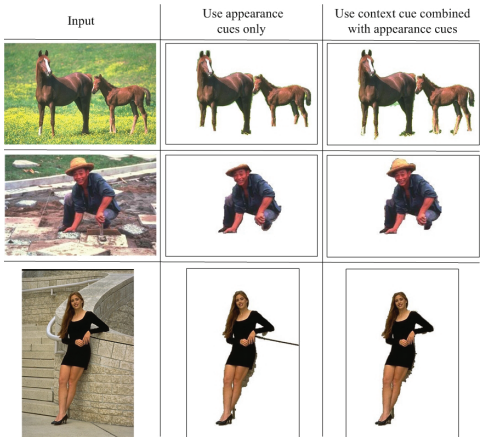

Figure 6. Left column: input images. Middle column: the results obtained by our method that uses appearance cue only. Right column: the results obtained by our method that integrates both appearance and contextual cues.

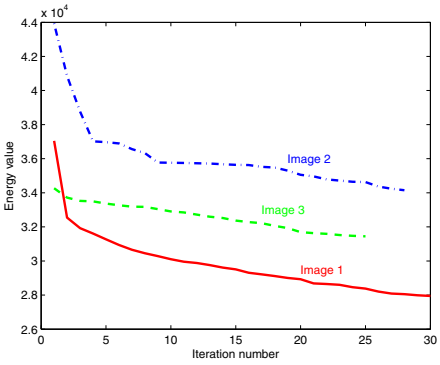

Figure 7. Energy values in the iterative process of energy minimization tested on the 3 images in Fig. 6.

scores.html. The total F-measure score of our algorithm is 0.91 ± 0.013, which significantly outperforms all other 5 algorithms tested. The F-score of our approach without using the auto-context cue is 0.88 ± 0.011, which is lower than using the auto-context cue, but is also higher than all other 5 algorithms. This strongly manifested the efficacy of the contextual cue and saliency cue leveraged.

Experiment 5. Subjective evaluation. Furthermore, we also compare our method with two recent interactive segmentation methods [16, 2]. To ensure the fairness of the comparison in terms of initialization, for GrabCut [16], we marked a bounding box exactly containing the object, and used the GrabCut implementation of *www.cs.cmu.edu/~mohitg/segmentation.htm*; for the unified approach [2], we only quoted the best results reported by the authors from *www.wisdom.weizmann.ac.il/~vision/GoodSegment.html*. Fig. 8 shows several results. The empirical results show that our automatic method compares favorably with these two interactive segmentation methods.

Algorithms	F-measure score	Remarks
Our method	0.91 ± 0.013	Automatic
Our method without using auto-context cue	0.88 ± 0.011	Automatic
Unified approach[2].	0.87 ± 0.01	Interactive
Cues integration[1].	0.86 ± 0.012	Automatic
Texture segmentation[8].	0.83 ± 0.016	Automatic
Normalized cut[18].	0.72 ± 0.018	Automatic
Meanshift[7].	0.57 ± 0.023	Automatic

Table 2. F-measures of comparing our method with 5 state-of-the-art segmentation algorithms by testing them on the Weizmann single object database[1].

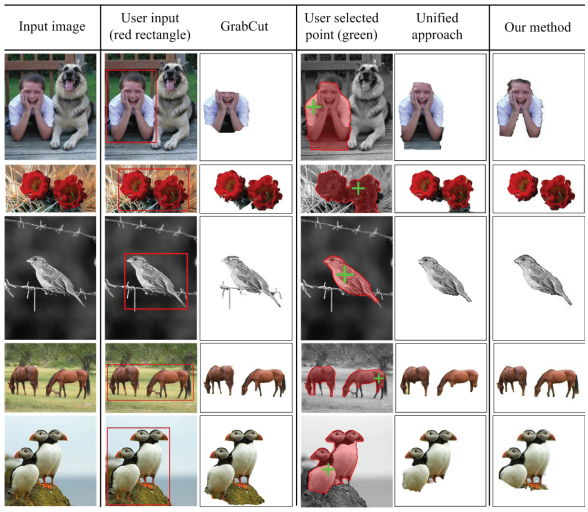

Figure 8. Comparison results of our method with two interactive segmentation methods proposed in [16, 2].

Experiment 6. Recognition results. Upon convergence of the iterative process of the energy minimization, besides the extracted salient object, we also obtained a fully trained auto-context classifier which can be readily used to recognize the same type of object in new images. Fig. 9 shows some recognition results obtained by applying the auto-context classifier learned from one image to new images. As it is clearly shown, the learned auto-context model generalized well to new images.

7. Conclusion

We present an automatic salient object extraction method. Our method is able to automatically extract the object of interest from its background without any user intervention. This is enabled by casting saliency cue, contextual cue and appearance cue into a unified energy minimization framework. Empirical results on four popular segmenta-

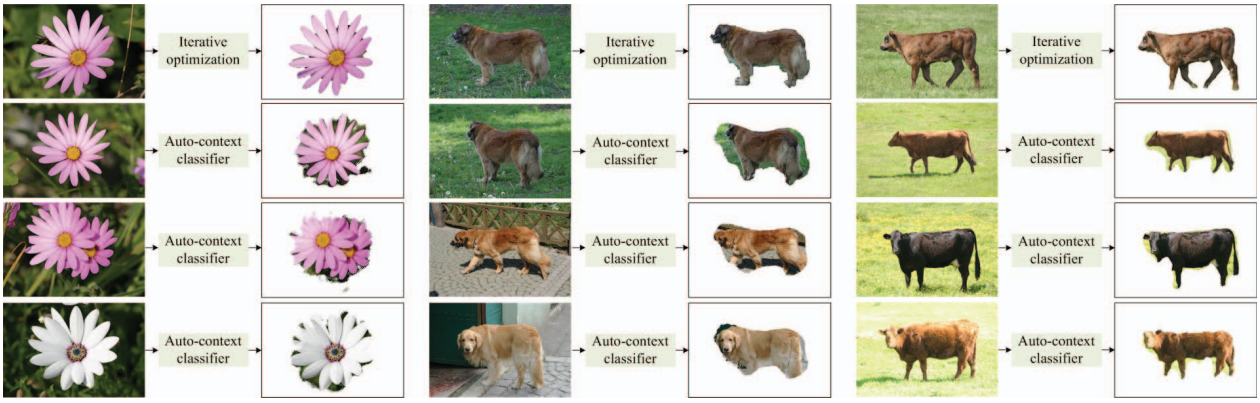

Figure 9. Recognition results. The 1st row are input images and the corresponding segmentation results. The rows 2, 3 and 4 are new images including the same type of objects and the corresponding recognition results.

tion benchmarks demonstrated the superb performance of our method. It compares favorably with even foreground extraction algorithms which leveraged user interaction. It shall be noted that the accuracy of the initial segmentation obtained from the bottom-up visual saliency model sometimes affects the result of our method, which is the focus of our future research.

Acknowledgements. This work was supported by NSFC under Grant No. 60875008 and Grant No. 90920301, and the China 973 Program under Grant No. 2010CB327902.

References

[1] S. Alpert, M. Galun, R. Basri, and A. Brandt. Image segmentation by probabilistic bottom-up aggregation and cue integration. In *CVPR*, 2007.

[2] S. Bagon, O. Boiman, and M. Irani. What is a good image segment? A unified approach to segment extraction. *ECCV*, 2008.

[3] S. Belongie, J. Malik, and J. Puzicha. Shape matching and object recognition using shape contexts. *TPAMI*, 2002.

[4] Y. Boykov and G. Funka-Lea. Graph cuts and efficient nd image segmentation. *IJCV*, 2006.

[5] Y. Boykov and M. Jolly. Interactive graph cuts for optimal boundary & region segmentation of objects in ND images. In *ICCV*, 2001.

[6] Y. Boykov and V. Kolmogorov. An experimental comparison of min-cut/max-flow algorithms for energy minimization in vision. *TPAMI*, 2004.

[7] D. Comaniciu and P. Meer. Mean shift: A robust approach toward feature space analysis. *TPAMI*, 2002.

[8] M. Galun, E. Sharon, R. Basri, and A. Brandt. Texture segmentation by multiscale aggregation of filter responses and shape elements. In *ICCV*, 2008.

[9] S. Gould, J. Rodgers, D. Cohen, G. Elidan, and D. Koller. Multi-class segmentation with relative location prior. *IJCV*, 2008.

[10] J. Harel, C. Koch, and P. Perona. Graph-based visual saliency. *NIPS*, 2007.

[11] G. Hua, Z. Liu, Z. Zhang, and Y. Wu. Iterative local-global energy minimization for automatic extraction of objects of interest. *TPAMI*, 2006.

[12] Y. Li, J. Sun, C. Tang, and H. Shum. Lazy snapping. *ACM Transactions on Graphics*, 2004.

[13] T. Liu, Z. Yuan, J. Sun, J. Wang, N. Zheng, X. Tang, and H. Shum. Learning to detect a salient object. *TPAMI*, 2010.

[14] D. Martin, C. Fowlkes, and J. Malik. Learning to detect natural image boundaries using local brightness, color, and texture cues. *TPAMI*, 2004.

[15] A. Rabinovich, A. Vedaldi, C. Galleguillos, E. Wiewiora, and S. Belongie. Objects in context. In *ICCV*, 2007.

[16] C. Rother, V. Kolmogorov, and A. Blake. Grabcut: Interactive foreground extraction using iterated graph cuts. *ACM Transactions on Graphics*, 2004.

[17] S. Savarese, J. Winn, and A. Criminisi. Discriminative object class models of appearance and shape by correlatons. *CVPR*, 2006.

[18] J. Shi and J. Malik. Normalized cuts and image segmentation. *TPAMI*, 2002.

[19] J. Shotton, J. Winn, C. Rother, and A. Criminisi. Textonboost: Joint appearance, shape and context modeling for multi-class object recognition and segmentation. *ECCV*, 2006.

[20] Z. Tu. Auto-context and its application to high-level vision tasks. In *CVPR*, 2008.

[21] O. Veksler. Star shape prior for graph-cut image segmentation. *ECCV*, 2008.

[22] M. J. Wainwright, T. S. Jaakkola, and A. S. Willsky. Tree-reweighted belief propagation algorithms and approximate ml estimation by pseudo-moment matching. In *Workshop on Artificial Intelligence and Statistics*, 2003.

[23] D. Walther and C. Koch. Modeling attention to salient proto-objects. *Neural Networks*, 2006.

[24] S. Wu and F. Crestani. Data fusion with estimated weights. In *ICIKM*, 2002.

[25] J. Yedidia, W. Freeman, and Y. Weiss. Generalized belief propagation. In *NIPS*, 2000.