Fine-Grained Crowdsourcing for Fine-Grained Recognition

Jia Deng, Jonathan Krause, Li Fei-Fei
Computer Science Department, Stanford University

Abstract

Fine-grained recognition concerns categorization at sub-ordinate levels, where the distinction between object classes is highly local. Compared to basic level recognition, fine-grained categorization can be more challenging as there are in general less data and fewer discriminative features. This necessitates the use of stronger prior for feature selection. In this work, we include humans in the loop to help computers select discriminative features. We introduce a novel online game called "Bubbles" that reveals discriminative features humans use. The player's goal is to identify the category of a heavily blurred image. During the game, the player can choose to reveal full details of circular regions ("bubbles"), with a certain penalty. With proper setup the game generates discriminative bubbles with assured quality. We next propose the "BubbleBank" algorithm that uses the human selected bubbles to improve machine recognition performance. Experiments demonstrate that our approach yields large improvements over the previous state of the art on challenging benchmarks.

1. Introduction

Fine-grained recognition concerns recognizing sub-ordinate object classes. Examples include distinguishing different breeds of dogs, species of birds, models of cars, and categories of mushrooms. These tasks yield a great deal of information for a human user and can thus add tremendous value to society.

Fine-grained recognition is challenging. There is in general limited data as fine grained labels are much harder to acquire. More importantly, there are much fewer discriminative features compared to categorization at the basic level. Distinguishing a dog and a microwave is easy because there are plenty of helpful visual cues. In comparison, the difference between fine grained classes can be very subtle and only a few key features matter. Consider, for example, two very similar woodpeckers "Northern Flicker" and "Red Bellied Woodpecker" (bird A and B in Figure 1). If irrelevant features are used, it is virtually impossible to distinguish the two. But if we know the key differences, the task becomes easy — bird A has a spotted chest while the top of bird B's head is red. Given limited data, automatic selection of discriminative features becomes especially difficult as a large number of irrelevant features can cause severe overfitting.

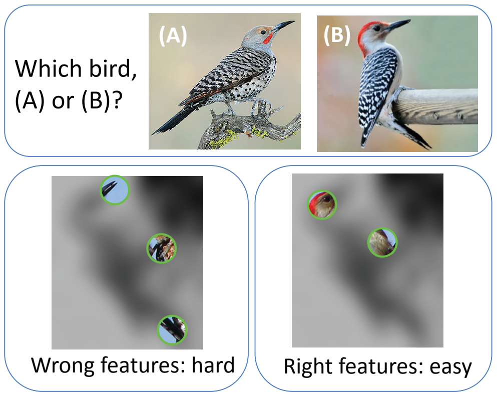


Figure 1. The distinction between fine-grained categories is often very subtle. It is crucial to identify the key features – if the wrong features are selected, the task can be very difficult. A small number of right features, on the other hand, makes the task easy.

To tackle the challenge of feature selection, one approach is applying specialized domain knowledge, This approach can yield great success [16], but demands from the researcher a deep understanding of the specific domain. Another promising direction is including the crowd in the loop by having humans either label or propose parts and attributes [3, 13, 11, 22, 9, 21]. These approaches can potentially reduce the burden of domain specific engineering. We refer to this category of approaches "fine-grained crowdsourcing". It is "fine-grained" in two senses: (1) the crowd not only provides class labels indicating *what* the object is, but also provides detailed information on *how* humans achieve fine grained recognition; (2) the learning algorithm not only optimizes the classification accuracy but also incorporates the "finer-grained" hints from the crowd, which would help avoid overfitting and lead to better generalization performance. The challenge, however, lies in how to design effective annotation tasks. Existing approaches either ask humans to label pre-defined parts and attributes [3, 13], or assign open-ended tasks such as proposing semantic labels [20, 11, 22, 9] and specifying visual primitives (keypoints or regions) [9, 21]. Labeling parts and attributes places a large burden on the researcher in specifying the annotation task. The open-ended alternatives, on the other hand, can be difficult for quality control as the tasks are highly subjective.

In this paper we take one step further in this direction by introducing a novel crowdsourcing approach to help computers select discriminative features. This approach does not require the researcher to specify parts and attributes, is open-ended, and has automatic quality control. Specifically, we propose a novel online game called "Bubbles" that reveals the discriminative features. Consider bird species identification as an example. At each round of the game, a player sees example images for two bird species. She is then given a new image and is asked to classify the bird into one of the two species. She earns points for correct identification and loses points otherwise. Regardless of the outcome, the game advances to the next round with a new image and possibly a new pair of bird species. The key twist of the game is that the new image is always heavily blurred so that the player can only see a rough outline of the bird. The player can, however, click to reveal small, circular areas of the image ("bubbles") to inspect the full details, with a penalty on game points. Through proper setup of reward, the game can guarantee that bubbles selected by a successful human player contain discriminative features.

The game enjoys the following advantages: (1) *Domain agnostic*. The only assumption is that humans can discover discriminative visual features from a handful of examples. Thus it applies to a wide range of domains and appeals to a generic crowd. In fact, learning to tell unfamiliar categories apart under time pressure creates challenges and fun. (2) *Automatic quality assurance*. If the players earns high scores, we know with certainty that the areas chosen must be important. (3) *Cost effective*. The game provides entertainment and people will volunteer to play. This can enable large scale data collection with low or zero cost.

Our second contribution is "BubbleBank", a new algorithm that uses the crowd-selected bubbles for fine-grained recognition. For each bubble from the game, we generate a "bubble detector" that tries to detect the same pattern from other images. Each image can then be represented by "BubbleBank", a collection of max-pooled responses from each bubble detector. We demonstrate that BubbleBank can improve previous state of the art methods by large margins on challenging fine-grained benchmarks. Fig. 2 illustrates our complete framework.

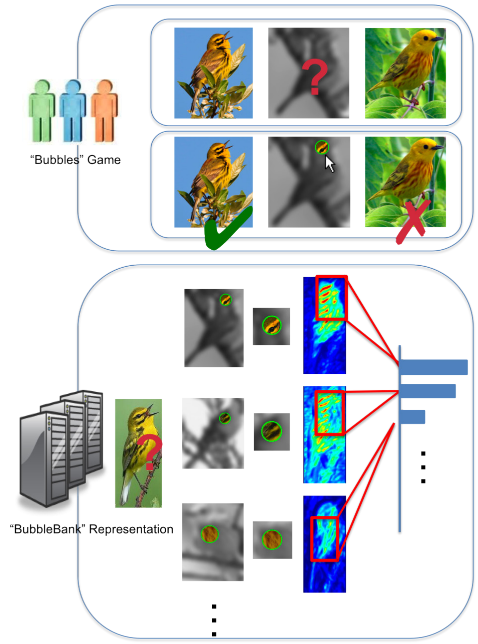


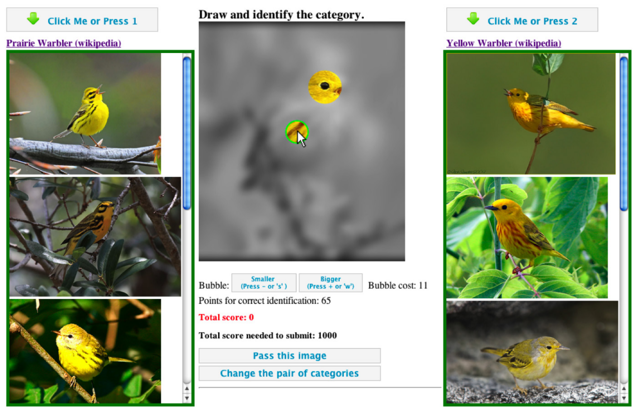
Figure 2. In our approach, the crowd first plays the "Bubbles" game, trying to classify a blurred image into one of the two given categories. During the game, the crowd is allowed to inspect circular regions ("bubbles"), with a penalty of game points. In this process, discriminative regions are revealed. Next, when a computer tries to recognize fine grained categories, it collects the human selected bubbles and detects similar patterns on a image. The detection responses are max-pooled to form a "BubbleBank" representation that can be used for learning classifiers.

2. Related Work

Our work shares the same end goal as existing work on fine grained recognition [16, 6, 15, 25, 19, 37, 36], but our approach is more aligned with the general line of research that places humans in the loop [20, 11, 22, 29, 9, 5, 24, 23, 20, 32, 4, 26]. In particular, there has been success in seeking to understand *how* humans perform recognition, *e.g.* by asking humans to directly provide annotation rationales [9], to label features in NLP tasks [10], to describe the differences between pairs of images [20], or to perform tasks that are parts of the machine pipeline [23]. Our work is different in that we use online games to discover discriminative features for fine grained recognition.

This work relates to many human vision studies. The game is named after a well known psychology technique for studying features that humans use for face recognition [14]. Human subjects are shown a face image with random bubbles revealed and asked to identify the gender or expression. Our approach differs in that our bubbles are actively chosen by the player. Another connection to human vision studies is that our game to a certain extent resembles eye tracking, revealing the locations looked at by humans.

Our game also draws inspiration from human computation [30, 31, 17], especially the seminal "Peekaboom" game [31]. In this two player game, player A is given a word (*e.g.* "cow") and an image. Player A can then click to reveal parts of the image to Player B. Player B needs to type the word "cow" after seeing only the revealed area. Our game is different. First, Peekaboom is not suitable for fine grained


Click Me or Press 1
Prairie Warbler (wikipedia)
Draw and identify the category.
Bubble:
Smaller (Press - or 's')
Bigger (Press + or 'w')
Pass this image
Change the pair of categories
Click Me or Press 2
Yellow Warbler (wikipedia)

Bubble cost: 11
Points for correct identification: 65
Total score: 0
Total score needed to submit: 1000

Figure 3. The game UI. The goal is to correctly classify the center image into one of the two categories. A green bubble follows the cursor. The player can click to reveal the area inside the bubble. The more bubbles used, the fewer points the player can earn.

recognition because an average player cannot be expected to come up with the same word "Northern Flicker". Second, the goal of Peekaboom is to locate the objects, not the discriminative parts. In particular, what parts are discriminative for a category depends on what the reference category is. In our game, we replace word typing with binary choices and make discovering discriminative visual features between unfamiliar categories part of the game play. Another difference is that our game is for a single-player. This eliminates the need to match two players in real time, making it much easier to deploy on paid crowdsourcing platforms such as Amazon Mechanical Turk (AMT).

Finally, our BubbleBank algorithm is related to work that uses collections of part/object detectors [3, 13, 18]. Our approach differs in that our BubbleBank consists of detectors tailored to the outputs of the Bubbles game, with a simple representation that requires no additional detector learning.

3. The Bubbles Game

The Game Mechanism Fig. 3 shows the game UI. A player is given example images of two categories. In the center lies a a blurred, de-saturated image with only the rough outline of the object visible. The goal is to correctly classify the center image into one of the two categories. A green "bubble" (size adjustable) follows the mouse cursor as the player hovers over the center image. When the player clicks, the area under the circle is revealed in full detail. If the player answers correctly, she earns new points. Otherwise she loses points. Either way, the game then advances to the next round, with a new center image and possibly a new pair of categories. Note that all images are assumed to have ground truth class labels so that we can instantly judge the player's answers.

We design the reward of the game such that a player can only earn high scores if she identifies the categories correctly *and* uses bubbles parsimoniously. First, we set the penalty on wrong answers very large, for example, 100 points for correct identification but -300 for incorrect ones. This renders random guessing an ineffective strategy. Also, the player is allowed to pass difficult images or categories with no penalty, such that they are not forced to guess. Second, there is a cost associated with the total area revealed. The points for correct identification will decrease as more area is revealed. For example, in our experiments the scores typically drop to zero when about 30% of the object bounding box is revealed. This thus encourages careful bubble use. This reward setup therefore reliably distinguishes good players and assures the quality of their bubbles.

Another issue of game design is determining the amount of blurring for the center image. With insufficient blurring, the player can directly identify the category, whereas too much blurring would obscure the global shape. To address this issue, we start with a small amount of blurring and increase it gradually in new games until the use of bubbles becomes necessary. Note that this in fact creates useful side information about the scale of the discriminative features.

The game can be enjoyable as it has an engaging challenge-reward setup with instant feedback. To earn high scores, the user needs to discover the differences between highly confusing categories. This is similar to the classical "spot-the-difference" games. Next, the user needs to think about where to place the bubbles. To further enhance the experience, we can create a sense of time pressure by adding a countdown timer and "freezing" the bubbles for a few seconds once a certain amount of area has been revealed.

We finally note that there is nothing specific about birds in the game design. In particular, the players do not need to understand any attributes or parts. Thus the game can be readily applied in a different domain. The only assumption is that humans can learn from a few examples, which turns out to be valid through our large scale AMT deployment.

AMT Deployment The game is suitable for deployment on paid crowdsourcing platforms such as AMT. Each AMT task would consist of multiple rounds of games. The worker must score enough points in order to submit the task, otherwise the games will continue indefinitely. The threshold for submission is set high enough such that random guessing is infeasible. This ensures that only the good workers would be able to submit. Notably, there is no need to make approval/reject decisions, as is necessary for conventional tasks. All submissions are guaranteed to be high quality and can be automatically approved. This is a significant advantage as quality control is often a significant concern for crowdsourcing.

We deployed the game on AMT using the CUB-200-2010 bird dataset [35] that contains 200 types of birds. A total of 275 workers submitted 3339 tasks, with an average price of $0.07 each. This gives 90659 rounds of games, an

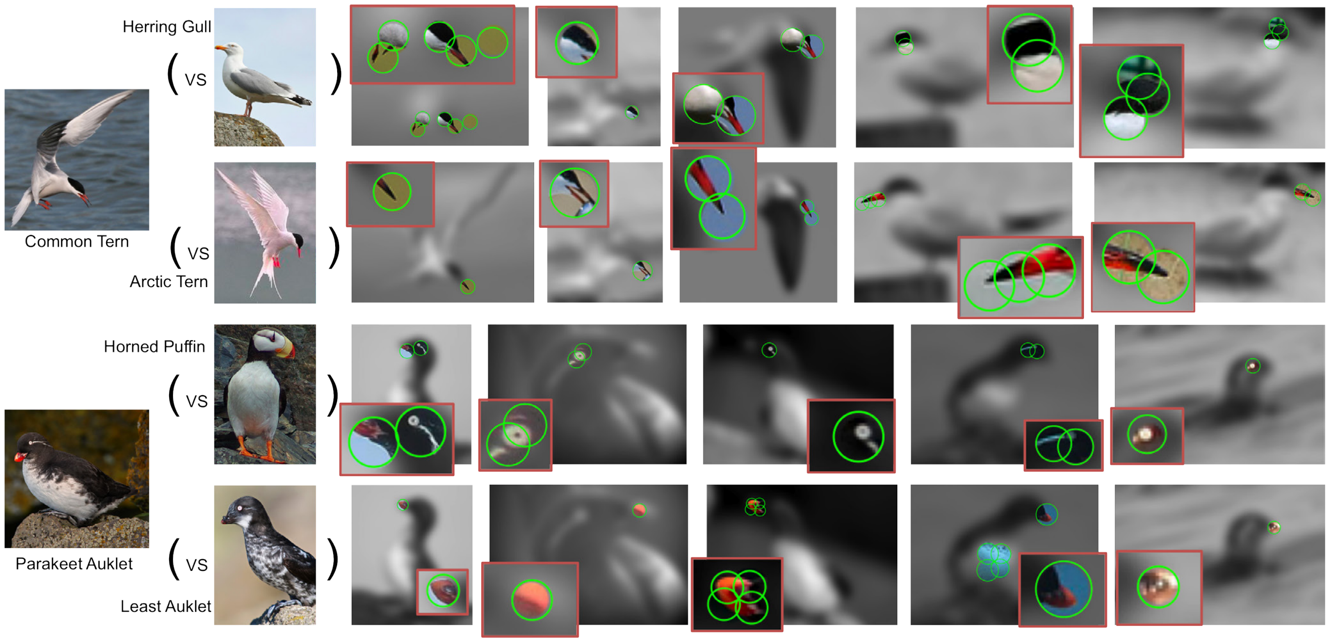


Figure 4. Examples of game results from AMT. The red boxes show zoomed-in views of the bubbles. **Top row:** Bubbles drawn on images of "Common Tern"' when compared against "Herring Gull". **Second row:** Bubbles for "Common Tern" on the same images of the top row when compared against "Arctic Tern". **Third row:** Bubbles for "Parakeet Auklet" when compared against "Horned Puffin". **Fourth row:** Bubbles for "Parakeet Auklet" on the same images of the third row when compared against "Least Auklet".

average of 27 rounds per task. We generate the games from visually confusing category pairs (see Sec. 5.2 for details). Each round identifies one image and lasts 25 seconds on average. Among all game rounds, 71% were successful (*i.e.* the player correctly identifies the category), 14% failed, and 15% were skipped by passing the image or switching categories. Fig. 4 shows examples of successful games for four pairs of categories. Remarkably, the workers are able to discover the subtle differences between very difficult pairs of categories. As in Fig. 4, the difference between "Common Tern" and "Arctic Tern" lies in whether the tip of the beak is black and in the length of the tail. Also observe how different features are selected for the same image when it is discriminated against different categories. When "Common Tern" is compared against "Herring Gull", the black patch on the head is discriminative and gets picked often. But when discriminated against "Arctic Tern", the black patch is no longer relevant and is less frequently chosen. For failed games, we observe that a significant fraction is due to a few very difficult category pairs.

It is also remarkable how little is needed to distinguish a pair of fine-grained categories. Fig. 5 plots the cumulative distribution of the area revealed in successful games — over 90% of the games reveal less than 10% of the object bounding box. This validates our hypothesis that (1) humans can indeed discover the fine differences from a handful of examples and (2) for fine-grained recognition, the key features are highly local.

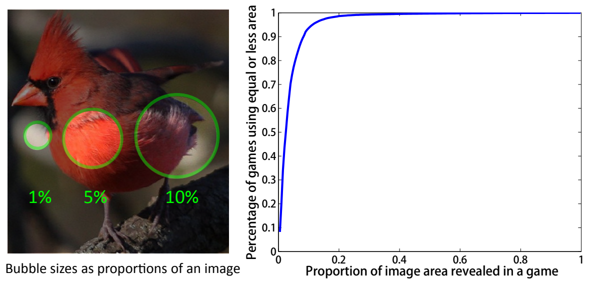


Figure 5. Statistics of image area revealed in successful games. The area revealed in most of the successful games is small. Over 90% of the games use less than 10% of the object bounding box.

Finally, we can aggregate the bubbles on the same image from multiple games played by multiple players and obtain a heat map of discriminative regions. Fig. 6 shows two examples. It suggests that the game can indeed discover meaningful cues for fine-grained recognition.

4. The BubbleBank Algorithm

The Bubbles game reveals discriminative features. In this section we show how to use the human selected bubbles to improve recognition. Our basic idea is to generate a detector for each bubble and represent each image as a collection of responses of the bubble detectors.

The Bubble Detectors Since each bubble is drawn in the context of discriminating two classes, we start by assuming

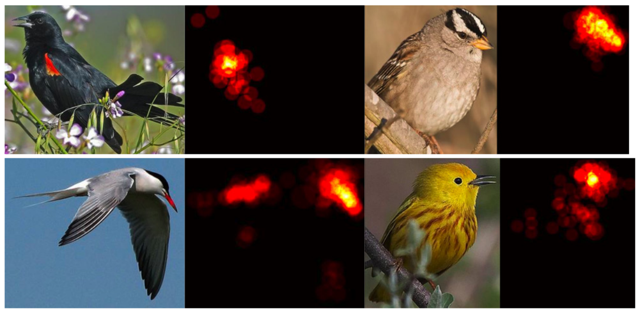


Figure 6. Heat maps of bubbles averaged over multiple games played by multiple players.

only two classes. Our intuition is that since each bubble contains discriminative features for recognition, it suffices to detect such patterns in a test image. It is thus natural to obtain a detector for each bubble.

How do we represent each bubble detector? Since each bubble is usually a small area, it can be represented by a single descriptor such as SIFT, or a concatenation of simple descriptors. This descriptor acts as an image filter — to detect on a test image, we convolve it with densely sampled patches and then take the maximum response (max-pooling). To further exploit the cues provided by the bubbles, we specify a pooling region for each detector. Instead of convolving with the entire image, each detector operates on a fixed, rectangular region whose center is determined by the relative location of the bubble in the original image. In other words, we have a strong spatial prior about where we expect to detect bubbles. Note that here we have assumed that the object has been localized, as is standard in the classification task in fine grained recognition [35, 37, 36].

Now, assume that we have collected multiple bubbles, each from a training image of one of the two classes (each training image can have multiple bubbles from a single round of game or multiple games played by different players). We can then form a bank of bubble detectors ("BubbleBank") and represent the image by a vector of the max-pooled responses from each detector, in a spirit similar to the ObjectBank [18] representation. Then a binary classifier can be learned on top of this representation. Fig. 2 illustrates the BubbleBank representation.

Extending to Multiple Classes Extending to multiple classes is straightforward — we can simply obtain bubbles for all pairs of categories and then use all of them to form our the BubbleBank. This, however, does not scale well with the number of classes because we need to run $O(K^2)$ games for K classes. Fortunately, obtaining bubbles for every pair of categories is unnecessary in practice. Not all classes are equally similar to others. It is likely that a bubble useful for differentiating a class from another very confusing class is also helpful for discriminating the same class against less similar ones. For example, the bubbles selected for "Common Tern" against "Herring Gull" in Fig. 4 are also useful for distinguishing "Common Tern" from the woodpeckers in Fig. 1. Therefore, for a large number of classes, we can pick only the most confusing category pairs. Specifically, we can first train a baseline classifier and then find out the confusing pairs via cross-validation. Alternatively, if a semantic hierarchy is available and visual similarity between classes is known to align well with the semantic hierarchy, as is often the case [8], we can directly select pairs of categories from within small subtrees.

We conclude this section by further comparing BubbleBank with related methods. On one hand, BubbleBank is related to a class of methods that learn attributes, parts, or object detectors (ObjectBank [18], Poselet [3], Birdlet [13]) and use their responses for classification. However, all these methods require additional annotation to train the detectors. On the other hand, BubbleBank is also related to more generic methods such as the codebook-free and annotation-free approach (CFAF) [36] and LLC [34]. These approaches use simple template representations but generate them through uniform or random sampling, with no additional supervision. Here we highlight some key differences: (1) our detectors are derived from a game that guarantees quality; (2) due to the assured quality, our representation of bubble detectors can be made very simple using low level descriptors without additional training; (3) we assume a strong spatial prior for each bubble detector.

5. Experiments

5.1. Dataset and Implementation

Dataset We use a standard fine-grained benchmark, the CUB-200 dataset [35] that contains 200 bird species. There are 6033 images in total and around 30 images per class. All of our experiments use the default training-test split. We experiment on the full dataset as well as a subset of 14 classes from the Vireo and Woodpecker family (CUB-14) that have been used in previous work [13, 36, 38]. All images are cropped to the bounding boxes, as is standard for many published results [35, 5, 37, 36, 1]. At test time, we do not use any ground truth information other than assuming that the image has been cropped.

Bubble Detectors We implement the bubble detectors using SIFT [27] and color histograms extracted at the bubble locations. The color histograms are based on a color naming method [28] that converts each pixel into a 11 dimensional vector, each dimension representing the probability of one of the 11 basic color terms (*e.g.* "black", "blue", "brown" *etc.*). We form an L_2 normalized histogram by averaging the color naming vectors within each bubble. The color vector is then concatenated with the SIFT descriptor to form the final 139-dimensional descriptor. To run the bubble detectors, we resize an image to a max dimension of 300 pixels

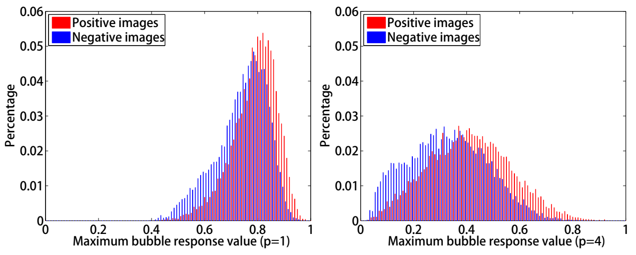


Figure 7. The effect of power scaling. When the max-pooled detector responses are raised to the power of $p > 1$, the differences between values in the higher range are amplified.

and extract the same SIFT and color descriptors on dense patches at every 2 pixels at multiple resolutions. The detector response at each location is the dot product of the image patch descriptor and the bubble descriptor. We specify the pooling region for each detector to be a 0.5×0.5 rectangle centered at the original bubble location, after normalizing all (x, y) coordinates to be in $[0, 1] \times [0, 1]$.

Power Scaling We train 1-vs-all linear SVMs [12] on top of our BubbleBank representations due to its superior efficiency. We found that raising each dimension to a constant power $p > 1$ (followed by L_2 normalization) can significantly improve the recognition performance. Specifically, we let $x_i \leftarrow x_i^p$ for dimension i of the BubbleBank vector. We determine p through cross-validation ($p = 8$ for CUB-14 and $p = 4$ for CUB-200). The improvements are large: setting $p = 8$ improves the mean average precision (mAP) of CUB-14 from 50.80% to 58.47%.

This power scaling technique is motivated by our observation on the values of detector responses. If a similar pattern exists in the image, then the detector will tend to have high response typically within $[0.8, 1]$ (assuming L_2 normalized descriptors); if no similar pattern exists, the maximum response, however, can take a wide range of values in $[0, 0.8]$. This is less desirable because the exact response value is unimportant when the detector does not fire. Power scaling thus serves to "suppress" the lower range and amplify the higher range. Fig. 7 (left) plots the distribution of the maximum bubble detector responses on images from a pair of classes in CUB-14. The red bars correspond to the responses of bubbles on images from the same class (*i.e.* positive examples) and blue bars the other class (*i.e.* negative examples). Observe the values are heavily biased toward the upper range. There is only a small difference between the modes of the two classes (around 0.82 versus around 0.78). After power scaling with $p = 4$, as shown in Fig. 7 (right), the ratio between the modes is amplified and the values in the lower range compressed.

5.2. Results

CUB-14 We first evaluate our approach on CUB-14. Following [36], both the training and testing sets are augmented by horizontally flipping the images. Since the 14 classes come from two visually very distinctive subgroups, vireo and woodpecker, we run the bubbles game within each subgroup. This gives 42 pairs of classes. We obtained 16336 bubbles from 4101 successful, positively scored games using a total of 210 unflipped training images. The same bubbles can be mirrored on the flipped images, which gives a total of 32672 bubble detectors.

Method	mAP (%)
MKL [5]	37.02
Birdlet [13]	40.25
CFAF [36]	44.73
Ours (SPM 1×1)	52.98
Ours (SPM $1 \times 1, 2 \times 2$)	48.63
Ours (Random Bubbles)	43.72
Ours	**58.47**

Table 1. Results on CUB-14.

Table 1 reports mean average precision of our method on CUB-14. We also compare with other methods including multiple kernel learning [5], Birdlet [13] and CFAF [36]. [1] Our method achieves 58.47%, outperforming the previous best result 44.73% [36] by a margin of 14%. Also note that Birdlet [13] requires additional annotations in the form of 2D key points and 3D ellipsoids. Our annotation task is much simpler but as effective.

How much does our fine-grained crowdsourcing matter? As a control experiment, we replace the crowdsourced bubbles with randomly generated ones while keeping everything else exactly the same. With random bubbles, the performance drops drastically, from 58.47% to 43.72%. Interestingly, with random bubbles, the performance is similar to 44.73% achieved by CFAF [36], which also uses random templates but further boosts performance by a bagging technique. This control experiment demonstrates that (1) the Bubbles game is essential and (2) the quality of the bubbles are indeed assured by the game mechanism.

We also evaluate the strong spatial prior used in our pooling. Table 1 reports the results of pooling over the entire image (SPM 1×1) and pooling over multiple regions from a $1 \times 1, 2 \times 2$ spatial pyramid. We see that both are significantly worse than pooling over a single, local neighborhood.

How many bubbles do we need? Fig. 8 reports recognition performances using subsampled bubbles (using 1%, 5%, 10%, 20%, 50%, 80% of the full set of 32672 bubbles). As a comparison, we also include the performances of using random bubbles. Strikingly, using only 1634 human selected bubbles (5% of the entire set), we already outperform CFAF [36] (51.05% versus 44.73%). Note that CFAF used 42000 random templates, a feature dimensionality of 882000, a customized SVM learning procedure, and

[1]The results from [38] on the same 14 classes are not comparable as a newer version [33] of the dataset with more images is used

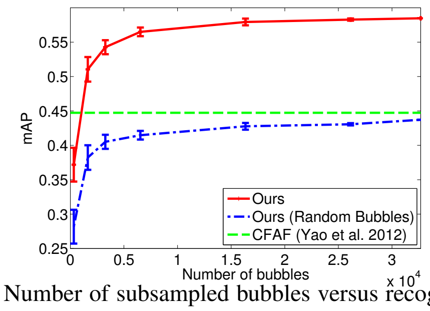


Figure 8. Number of subsampled bubbles versus recognition performance (mAP) on CUB-14. Our approach outperforms the previous state of the art [36] with only 1634 bubbles. Error bars are obtained by 10 runs.

80 rounds of bagging, whereas we achieve better performance using only 1634 bubbles, a feature dimensionality of 1634, off-the-shelf SVM training [12], and no bagging. The simplicity of our BubbleBank algorithm further confirms the effectiveness of our fine-grained crowdsourcing.

Fig. 9 shows success and failure cases of classification along with the top bubbles contributing to the predictions, We observe that the correct predictions can indeed be attributed to discriminative bubbles. The failure cases are also noteworthy. The first two cases are a result of treating each bubble independently. Often humans draw multiple bubbles on the same image and the bubbles may not be sufficiently discriminative in isolation. This suggests future work on detecting groups of bubbles. The third case is due to misfiring on the background. This points to a potential improvement through segmentation or through modeling context.

CUB-200 We next evaluate our approach on the full CUB-200 dataset using the standard setting [5, 35]. Since there are many more classes than CUB-14 and visually confusing pairs of classes are not necessarily from the same subgroup, we use a different approach to select the pairs for crowdsourcing. We first obtain the confusion matrix of the KDES method [2] via cross-validation on training data (no test data is used) and then pick the top 763 most confusing pairs. Each of the 200 classes shows up at least once in those 763 pairs. We then obtain 220242 bubbles through 46958 successful, positively scored games using training images.

Table 2 compares the classification accuracy of our method to various published results. We achieve an accuracy of 32.8%, a 6% improvement over the previous best result (26.7%) from TriCos [6] [2]. We also observe that random bubbles causes a large performance drop (from 32.8% to 26.5%), which again underscores the effectiveness of our fine-grained crowdsourcing.

Method	Accuracy (%)
MKL [5]	19.0
Random Forest [37]	19.2
Hierarchical Matching [7]	19.2
Multi-cue [15]	22.4
KDES [2, 1]	26.2
TriCos [6]	26.7
Ours (Random Bubbles)	26.5
Ours	**32.8**

Table 2. Results on CUB-200.

6. Conclusion

In this work, we have proposed a novel "human-in-the-loop" approach for fine-grained recognition. First, we introduce a new online game "Bubbles" that reveals important features humans in fine-grained recognition. The game is domain agnostic and guarantees high quality data. Second, we propose the "BubbleBank" algorithm that uses the human selected bubbles to learn classifiers for fine-grained categories. Experiments demonstrates large improvements of our approach over the previous state of the art.

[2][6] uses segmentation and only evaluates on uncropped images.

Acknowledgments We thank Alexandre Alahi, Michelle Greene, Olga Russakovsky, Bangpeng Yao, and anonymous reviewers for their comments. The research is partially supported by grants from Intel ISTC, an ONR-MURI, and NSF-IIS-1115313.

References

[1] http://www.cs.washington.edu/robotics/projects/kdes/.

[2] L. Bo, X. Ren, and D. Fox. Kernel descriptors for visual recognition. *NIPS*, 7, 2010.

[3] L. Bourdev and J. Malik. Poselets: Body part detectors trained using 3d human pose annotations. In *CVPR*, 2009.

[4] S. Branson, P. Perona, and S. Belongie. Strong supervision from weak annotation: Interactive training of deformable part models. In *ICCV*, Barcelona, 2011.

[5] S. Branson, C. Wah, F. Schroff, B. Babenko, P. Welinder, P. Perona, and S. Belongie. Visual recognition with humans in the loop. *ECCV*, 2010.

[6] Y. Chai, E. Rahtu, V. Lempitsky, L. Van Gool, and A. Zisserman. Tricos: A tri-level class-discriminative co-segmentation method for image classification. In *ECCV*, 2012.

[7] Q. Chen, Z. Song, Y. Hua, Z. Huang, and S. Yan. Hierarchical matching with side information for image classification. In *CVPR*, 2012.

[8] J. Deng, A. Berg, K. Li, and L. Fei-Fei. What does classifying more than 10,000 image categories tell us? *ECCV*, 2010.

[9] J. Donahue and K. Grauman. Annotator rationales for visual recognition. In *ICCV*, 2011.

[10] G. Druck, B. Settles, and A. McCallum. Active learning by labeling features. In *EMNLP*, 2009.

[11] K. Duan, D. Parikh, D. Crandall, and K. Grauman. Discovering localized attributes for fine-grained recognition. In *CVPR*, 2012.

[12] R. Fan, K. Chang, C. Hsieh, X. Wang, and C. Lin. Liblinear: A library for large linear classification. *The Journal of Machine Learning Research*, 9:1871–1874, 2008.

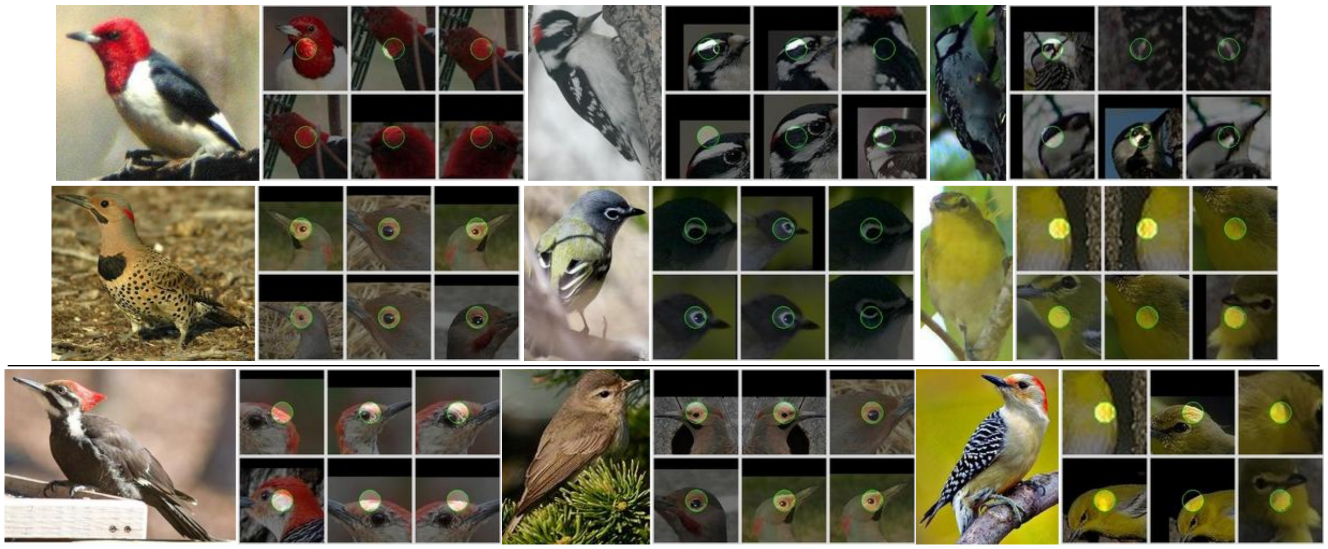


Figure 9. Test images and their top bubbles that contribute most to the classification decision. All bubbles are normalized to the same size for viewing. **Top and middle row:** Correctly classified test examples. **Bottom row:** Incorrectly classified test examples.

[13] R. Farrell, O. Oza, N. Zhang, V. Morariu, T. Darrell, and L. Davis. Birdlets: Subordinate categorization using volumetric primitives and pose-normalized appearance. In *ICCV*, 2011.

[14] F. Gosselin and P. Schyns. Bubbles: a technique to reveal the use of information in recognition tasks. *Vision research*, 41(17):2261–2271, 2001.

[15] F. Khan, J. van de Weijer, A. Bagdanov, and M. Vanrell. Portmanteau vocabularies for multi-cue image representation. NIPS, 2011.

[16] N. Kumar, P. Belhumeur, A. Biswas, D. Jacobs, W. Kress, I. Lopez, and J. Soares. Leafsnap: A computer vision system for automatic plant species identification. *ECCV*, 2012.

[17] E. Law, B. Settles, A. Snook, H. Surana, L. von Ahn, and T. Mitchell. Human computation for attribute and attribute value acquisition. In *Proceedings of the First Workshop on Fine-Grained Visual Categorization (FGVC)*, 2011.

[18] L. Li, H. Su, E. Xing, and L. Fei-Fei. Object bank: A high-level image representation for scene classification and semantic feature sparsification. *NIPS*, 24, 2010.

[19] J. Liu, A. Kanazawa, D. Jacobs, and P. Belhumeur. Dog breed classification using part localization. *ECCV*, 2012.

[20] S. Maji. Discovering a lexicon of parts and attributes. In *Second International Workshop on Parts and Attributes, ECCV*, 2012.

[21] S. Maji and G. Shakhnarovich. Part annotations via pairwise correspondence. In *Workshops at the Twenty-Sixth AAAI Conference on Artificial Intelligence*, 2012.

[22] D. Parikh and K. Grauman. Interactively building a discriminative vocabulary of nameable attributes. In *CVPR*, 2011.

[23] D. Parikh and C. Zitnick. Human-debugging of machines. In *Second Workshop on Computational Social Science and the Wisdom of Crowds, NIPS*, volume 11, 2011.

[24] A. Parkash and D. Parikh. Attributes for classifier feedback. In *ECCV*, 2012.

[25] O. Parkhi, A. Vedaldi, A. Zisserman, and C. Jawahar. Cats and dogs. In *CVPR*, 2012.

[26] A. Sorokin and D. Forsyth. Utility data annotation with amazon mechanical turk. In *CVPRW*, 2008.

[27] K. E. A. van de Sande, T. Gevers, and C. G. M. Snoek. Evaluating color descriptors for object and scene recognition. *TPAMI*, 32(9):1582–1596, 2010.

[28] J. Van De Weijer, C. Schmid, and J. Verbeek. Learning color names from real-world images. In *CVPR*, 2007.

[29] S. Vijayanarasimhan and K. Grauman. Large-scale live active learning: Training object detectors with crawled data and crowds. In *CVPR*, 2011.

[30] L. Von Ahn and L. Dabbish. Labeling images with a computer game. In *CHI*, pages 319–326. ACM, 2004.

[31] L. Von Ahn, R. Liu, and M. Blum. Peekaboom: a game for locating objects in images. In *CHI*. ACM, 2006.

[32] C. Wah, S. Branson, P. Perona, and S. Belongie. Multiclass recognition and part localization with humans in the loop. In *ICCV*, Barcelona, 2011.

[33] C. Wah, S. Branson, P. Welinder, P. Perona, S. Belongie, and C. Wah. Caltech-ucsd birds-200-2011, 2011.

[34] J. Wang, J. Yang, K. Yu, F. Lv, T. Huang, and Y. Gong. Locality-constrained linear coding for image classification. In *CVPR*, 2010.

[35] P. Welinder, S. Branson, T. Mita, C. Wah, F. Schroff, S. Belongie, and P. Perona. Caltech-ucsd birds 200. 2010.

[36] B. Yao, G. Bradski, and L. Fei-Fei. A codebook-free and annotation-free approach for fine-grained image categorization. In *CVPR*, 2012.

[37] B. Yao, A. Khosla, and L. Fei-Fei. Combining randomization and discrimination for fine-grained image categorization. In *CVPR*, 2011.

[38] N. Zhang, R. Farrell, and T. Darrell. Pose pooling kernels for sub-category recognition. In *CVPR*, 2012.