Support region estimation in biological images

Oscar Beijbom UCSD
9500 Gilman Drive
obeijbom@cs.ucsd.edu

Abstract

We describe a series of experiments relating to patch size selection in a challenging dataset of coral reef images and show that we can achieve significant performance increase over the naïve strategy of selecting a fixed patch size. Many computer vision methodologies use the bag of visual words representation of image objects. This is true, in particular, for the state of the art texture recognition frameworks. When applying these methods to recognition and segmentation of natural images, the issue of support region selection becomes crucial. We perform a series of experiments and conclude that the best strategy is to feed information aggregated over multiple support regions into a standard supervised machine learning technique.

1. Introduction

The bag of features representation is commonly used for computer vision tasks. In this domain an image patch is described using a histogram of visual word counts in that same patch. Given point of interest a fundamental question is how large neighborhood one should integrate over. We investigate this on a database of coral images using a texture recognition framework.

The basic assumption is that a larger integration area is better as the histogram of visual words will be more robust. Also, some textures are large by nature, and require a large integration area to be captured accurately. In some areas however, the points of interest fall on small image features, and integration area should be kept small to avoid pollution from surrounding classes. It is this conflict that is examined in this work. The images in fig. 1 illustrate the problem with some examples from a database of images of coral reef. Note, we do not address the related issues that occur on borders between textures.

2. Database

In this work we analyze image data collected from the island of Moorea in Tahiti as part of the Long Term Ecological Research (LTER) project [6] during 2008 and 2009. This project monitors coral reef sites at six locations around the island. In each location, four parts of the reef at different depths are sampled. The locations are shown in fig. 2. In this study we use data from the fringing reef, and the fore reef at 10 and 17 meter depth. The images from the lagoon backreef were not used in this paper as they are collected and annotated differently compared to the images.

Figure 1. Four different patches from the database, each with a region of 63 by 63 pixels indicated as a red box. fig. 1(a) shows a small algae patch. If the integration area in this image is too big, we will get more signal from the surrounding area than from the algae, indicating that we like to keep the area small. fig. 1(b) and fig. 1(c) show coral patches that both would benefit from a large area. In fig. 1(b) the indicated area might not be big enough to capture the larger textures, while in fig. 1(c) it is but the histogram of visual words would be more stable if a larger area is used. fig. 1(d) show a case where the point of interest is on the edge between classes. This issue is not addressed in this work.
we use. To achieve repeatability across years, a transect line is attached between points permanently anchored in the reef. Then, divers move along the transect line, acquiring images with a Nikon D-70 camera equipped with underwater strobes.

The camera is attached to a frame so that the distance from the ground is consistent, and thus the field of view is constant. Fig. 2(c) shows a sample image from the database. The 50 by 50 cm frame is seen in the image edges, and the white transect lines passes through the middle. The orange lines are cables attached to the frame to provide rigidity and a way for the diver to locate the center of the frame. With this acquisition method, the image locations differ slightly each year, but they are from the same part of the reef. This is sufficient for estimating overall coverage statistics of the major classes of coral and algae, which is the goal of this monitoring project.

The images are labeled using random point sampling. In this method, points (200) are overlaid on the images at random locations. Then, the scientist labels each point according to an established taxonomy for the site and ecological study. This methodology is commonly used throughout the community and is facilitated through software such as Coral Point Count [8]. In this study we use the eight major classes of coral and algae: (1) Rock / crustose coralline algae / Turf algae, (2) Macroalgae and (3) Sand. Five are coral genus: (4) Acropora, (5) Pavona, (6) Montipora, (7) Pocillopora, and (8) Porites. These eight classes account for 99% of the labels in the data set. The first class will be referred to as rock, and the second as Algae throughout this paper. Examples of annotated images are displayed in fig. ??.

This type of annotation demonstrates the complexity of the data. Unlike other commonly used data sets, objects do not have clear boundaries, and a bounding box is not indicative of the location. This poses a challenge in using standard recognition methods. We use 671 images acquired during April 2008 for these experiments.

It is worth mentioning the large within class variability in this data set. The Porites genus, for example, contains both encrusting and branching species. The class Macroalgae varies tremendously in shape and color, and often times ‘stick out’ from underneath the corals. Finally, some coral genus exhibit seasonal and depth variations. The rock category is often actually dead corals overgrown by crustose coralline algae, giving it a similar shape, but a different color and texture.

3. Multiple instance learning

In this section we briefly review the Multiple Instance Learning (MIL) [5] paradigm, with focus on Support Vector Machines (SVMs). In the MIL paradigm, a weaker assumption is made on the labeling information. The labels are provided in bags and a label is attached to the bag rather than the individual samples. The bags will be labelled as positive if at least one sample in the bag is positive, and negative if all samples are negative. The method has been used successfully in the face detection paradigm [13], where all bounding boxes from an image known to contain a face can be thrown together in a positive bag, and trained against a bag of negative samples. Andrews et.al [1] explored the opportunity to use SVMs in the MIL setting. They devices a simple optimization heuristic where one iterates between 1) training a SVM using a current set of positive labels against all labels, and 2) classifying all positive bags and using the instances that are classified as positive as positive examples in the next iteration. In the test phase, an instance gets classified as positive if the classifier decision value is above a certain threshold. The algorithm is described in detail in their paper, referred to as the ‘mi-SVM’ optimization heuristic.

4. Preprocessing

The objects we aim to classify are small, do not have clear boundaries, and are often mixed with other classes. Also, there is no clear sense of shape and it therefore seems appropriate to represent them using texture descriptors. We choose to model this using histogram of textons [10, 12]. In this section we describe the mapping of images pixels to visual words, or in this case, textons. We use a pipeline inspired, in large, by the method presented in [12].

Filter bank A key component to the method in [12] is the filter bank. We use the rotational invariant, Maximum Response (MR), filter bank also introduced in [12]. The filter bank for our choice of filter sizes is shown in fig. 3. The idea is to encode rotational invariance by first filtering with bar and edge filters of different orientations and then letting the maximum over the orientations be the output. It also contains a circular gaussian and laplacian filter. In this paper we use a filter bank that includes three bar and three edge filters of standard deviations 1, 3, 8 along the short direction and one set of the circular filters of standard deviation 3. This filter bank will thus output an 8 dimensional filter vector after taking the max over the orientations.

The method as presented in [12] does not give special care to color. Even though colors are not easily corrected for
in general, and in particular not underwater [2], we saw, empirically, that color information was indeed useful for this task. We encode color information by applying the filters to each color layer in a L*a*b* color space and stack the filter response vectors, resulting in a 24 dimensional filter response vector. The L*a*b* color space was empirically superior to greyscale, RGB, and HSV color spaces when performance was measured on the end to end system.

Dictionary The first step of the method pipeline is to create a dictionary of textons. For this we put aside a subset of the 2008 images that will not be used in any test sets. Filter responses from each of the eight classes are aggregated across the images, and k-means [7] clustering with 15 cluster centers is applied. Finally the cluster centers, or textons, from the different classes are merged to create the dictionary with 120 words, each 24 dimensions. This dictionary is used in all experiments in this paper.

Descriptors We use histograms of textons as descriptors throughout this paper. Given an image, we filter it as described in Sec. 4. Each filter response is then mapped to the texton with smallest 2-norm distance. This way an integer valued texton map is created. The actual feature vector, or descriptor, is simply the normalized histogram of textons over an area around a point of interest.

A sample image, along with filter responses and the texton map is shown in fig. 2.

Implementation Details All images are subsampled to roughly 1000 by 1000 pixels, converted to L*a*b* color space and intensity balanced for each color channel individually, by subtracting the mean intensity and dividing by the intensity standard deviation. Also, following [11, 9, 10] we apply a contrast normalization after filtering as

$$F(x) \leftarrow F(x) / \frac{\log(1 + L(x)/0.03)}{L(x)}$$

where $L(x) = \|F(x)\|_2$ is the magnitude of the 24 dimensional filter response vector, $F(x)$ at pixel $x$.

5. Experiments

In this section we address how to proceed with machine learning in this setting. Given a number of interest points and corresponding annotations we crop out a 221 by 221 patch around these points from the texton map created as detailed in Sec. 4. This number, 221, is chosen to be big enough to cover sufficient image information across all classes. Our data thus consists of integer valued images, $X_1, \ldots, X_n$, $X_i \in \mathbb{N}^{221 \times 221}$, and labels $y_1 \ldots y_n$, $y_i \in \{1, 2 \ldots 8\}$. We use two thirds of the images as train data.

Histogram Extraction Now consider algorithm 1 that takes as input on of these images, the number of textons and a patchsize. By calling this algorithm with different
values of $PS$ we extract histograms of textons from areas of different size around the center of the image patch. These can, in turn, be fed into a machine learning algorithm.

**Algorithm 1** Normalized histogram extraction

1. **input**: a texton map image, $TM$, a patch size, $PS$, and the number of textons, $NT$.
2. **output**: a normalized histogram, $h$.
3. $C = \text{floor}(\text{size}(TM, 1)/2)$ (this is the center row and column of the patch)
4. **for** $i = 1 : NT$ **do**
5. $h(i) = \sum_{row=C-PS}^{C+PS} \sum_{col=C-PS}^{C+PS} TM(row, col) == i$
6. **end for**
7. $h := \frac{\sum_{i} h}{n}$

**Learning methodology** Support Vector Machines [4] (SVM), with Radial Basis Function [4] kernel are used throughout this work. For each experiment the RBF-SVM training step is preceded by a 4-fold cross validation step on the train data, where parameters $\gamma$ and $C$ are optimized by a logarithmic grid search over values $\log \gamma \in \{-5, -4, \ldots, 4, 5\}$ and $\log C \in \{-5, -4, \ldots, 4, 5\}$. We use the LIBSVM [3] implementation for all experiments.

**Error function** To evaluate the results, the f-score, or harmonic mean, defined as

$$\text{f-score} = 2 * \frac{\text{precision} * \text{recall}}{\text{precision} + \text{recall}},$$

(2)

to evaluate the method accuracy. Precision and recall are here defined as

$$\text{precision} = \frac{1}{n} \sum_{i=1}^{n} \frac{tp_i}{tp_i + fp_i},$$

(3)

$$\text{recall} = \frac{1}{n} \sum_{i=1}^{n} \frac{tp_i}{tp_i + fn_i},$$

(4)

where $tp_i$, $fp_i$, $fn_i$ are the true positive, false positive and false negatives rates respectively for class $i$.

We are now ready to formulate the experiments.

1. **Fixed patch size**: This is the naive strategy in which we do a line search over patchsizes to find the best possible fixed patchsize. We try patchsizes, $PS = [10, 30, 60, 110]$. For each patchsize we call algorithm 1 to extract normalized histograms, and proceed with a standard SVM training and testing. Fig. 4 show these patchsizes drawn on a few locations in an image.

2. **All patch sizes at once**: Call algorithm 1 for each of the four patchsizes, and concatenate the featurevectors. Then proceed with standard SVM training.

3. **MIL**: Train a one-versus-all MIL SVM as described in Sec. 3. In the testphase take the following two step approach. 1) classify all four patchsizes with all one-versus-all classifiers, yielding 4 decision values per class. 2) label the test sample with the label of the classifier that achieved the highest score across patchsizes. This is the standard method as proposed in the MIL literature. We also use a normalized version of this heuristic. The reason is that the normalized histograms for the smaller patchsizes are noisier as they integrate over fewer samples, which might make the decision values ‘spike’ more than those from other classes and therefore get selected more often. To address this I explore a normalized test procedure as follows. 1) classify all four patchsizes with all one-versus-all classifiers, yielding 4 decision values per class. 2) normalize by the standard deviation of all decision values for each patch size (but across all samples and labels). 3) label the test sample with the label of the classifier that achieved the highest (normalized) score across classes.

4. **Fixed patch size train - MIL test**: Train a one-versus-all SVM using one patchsize. Then use the test heuristics described in item 3.

In summary, the first experiment is to be viewed as a baseline experiment, the second essentially puts its hope to the power of the data, and the last two tries to do something more sophisticated by applying the MIL framework.
6. Results

The results for each experiment is shown in table 1, and convey a clear message. Experiment 2, using all patch sizes at once, is the best strategy. We now go though and comment on each experiment individually.

Table 1. Result overview. The reported scores are 1-fscore, and thus a lower value is better.

<table>
<thead>
<tr>
<th>Experiment</th>
<th>Score</th>
</tr>
</thead>
<tbody>
<tr>
<td>Exp1, patchsize 10</td>
<td>.51</td>
</tr>
<tr>
<td>Exp1, patchsize 30</td>
<td>.36</td>
</tr>
<tr>
<td>Exp1, patchsize 60</td>
<td>.33</td>
</tr>
<tr>
<td>Exp1, patchsize 110</td>
<td>.36</td>
</tr>
<tr>
<td>Exp2</td>
<td>.27</td>
</tr>
<tr>
<td>Exp3, standard</td>
<td>.42</td>
</tr>
<tr>
<td>Exp3, normalized</td>
<td>.40</td>
</tr>
<tr>
<td>Exp4, standard</td>
<td>.44</td>
</tr>
<tr>
<td>Exp4, normalized</td>
<td>.42</td>
</tr>
</tbody>
</table>

6.1. Experiment 1

The results correspond fairly well with what can be expected after looking at fig. 4. The smallest patch size is by far the worst, when used on its own. The other yield similar performance, with the 121 by 121 patch being the best. From the confusion matrices (not shown here), we learn there is some variation between what is good for the different classes also. The smallest patchsize work fairly well for the algae class (which has a distinct signature, but is often found in small patches), while the coral classes follow the trend indicated by the f-scores. The confusion matrix for the best performing patchsize is shown in fig. 5.

6.2. Experiment 2

By concatenating the histograms from all patchsizes, we achieve by far the best results. The problem with this approach is that it provides little insight in what was the more effective patchsize. This will have to be found out by repeated testing of different combinations of patch sizes. The confusion matrix for this experiment is shown in fig. 6.

6.3. Experiment 3

It seems the MIL approach doesn’t work that well for this paradigm. However, looking at the meta data from the training procedure, it reveals something about the data. fig. 7 shows the distribution of samples used in each iteration of the MIL algorithm. The columns correspond to iteration 1 to 20, and the rows class 1 to 8. As is evident from the figure, the algorithm is initialized using all samples from the second smallest patchsize. This once again confirms that the algae class occur in smaller patches throughout the data base.

As far as then normalization strategy, is seems to improve things a little. However, the intuition that the smaller patchsize would yield ‘noisier’ histograms that were more prone to large variations in decision values, is wrong as indicated in fig. 8. The improved performance therefore likely stems from the fact the the normalization makes it so the smallest patchsize, which is the least reliable, is listened to less often.

6.4. Experiment 4

This, also, did not work better than the baseline setup, and even worse that the MIL setup. Some suggestion on
Figure 7. This shows the distribution of samples used in each iteration of the MIL algorithm. The columns correspond to iteration 1 to 20, and the rows class 1 to 8. As is evident from the figure, I initialize using all samples from the second smallest patchsize.

Figure 8. Histograms of which patchsize the largest decision value is picked from in experiment 3. The unnormalized setting is shown in fig. 8(a), and the normalized in fig. 8(b). Contrary to what was expected, the largest decision values do not come from the smallest patchsize, but are rather evenly distributed. However, those labels do have a larger standard deviation, so after normalization they get selected much more seldom.

7. Discussion

We examine four strategies that deal with the problem of finding a good integration area in biological images. The naive strategy of doing a line search to find the best patch size yields good results, but indicates that different classes are captured better with different patch sizes. The second strategy relies on the learner to figure out the best integration area for each interest point in the image. This method works well and improves performance quite significantly as is evident when comparing the confusion matrices in fig. 5 and fig. 6.

The last two strategies use some element of MIL. Experiment 3 uses a full MIL pipeline, and experiment 4 uses the MIL classification strategy only. These results are hard to explain, as I would expect them to outperform, at least, the fixed patch size strategies. One reason might be that the setting, strictly speaking, is not pure MIL. The reason is that, by the way we create the bags, positive samples might be in the negative bags. Consider the situation where we train a one-versus-all classifier for the *porites* class. We use all patch sizes from the points labelled as *porites* to create a positive bag per point. So far so good. For the negative bags, however, we use all patch sizes from all other points as negative bags. Let’s say there is a small patch of *algae* surrounded by a *porites* colony. The larger patches extracted around this point will contain mainly *porites* textons. The negative bag will therefore, in essence, contain a positive sample, something which violates the MIL strategy. Another problem, that is perhaps more realistic, is that the histograms extracted from patches of different sizes have so different characteristics that they can not be treated as IID samples, which violates a fundamental supervised learning assumption.

8. Conclusion

We investigate different strategies for interest area selection in texture based segmentation of biological images. Our results indicate that the best option is to select multiple areas, concatenate the information and feed this extended feature vector into a standard supervised machine learning engine. Doing this yield a significant classification performance on a holdout test set when compared to using a single, fixed patch size.

References


