Hierarchical classification with reject option for live fish recognition

Phoenix X. Huang
University of Edinburgh
Xuan.Huang@ed.ac.uk

Bastiaan J. Boom
University of Edinburgh
bboom@inf.ed.ac.uk

Robert B. Fisher
University of Edinburgh
rbf@inf.ed.ac.uk

Abstract

A live fish recognition system is needed in application scenarios where manual annotation is too expensive, *i.e.* too many underwater videos. We present a novel Balance-Enforced Optimized Tree with Reject option (BEOTR) for live fish recognition. It recognizes the top 15 common species of fish and detects new species in an unrestricted natural environment recorded by underwater cameras. The three main contributions of the paper are: (1) a novel hierarchical classification method suited for greatly unbalanced classes, (2) a novel classification-rejection method to clear up decisions and reject unknown classes, (3) an application of the classification method to free swimming fish. This system assists ecological surveillance research, *e.g.* fish population statistics in the open sea. BEOTR is automatically constructed based on inter-class similarities. Afterwards, trajectory voting is used to eliminate accumulated errors during hierarchical classification and, therefore, achieves better performance. We apply a Gaussian Mixture Model (GMM) and Bayes rule as a reject option after the hierarchical classification to evaluate the *posterior* probability of being a certain species to filter less confident decisions. The proposed BEOTR-based hierarchical classification method achieves significant improvements compared to state-of-the-art techniques on a live fish image dataset of 24150 manually labelled images from south Taiwan sea.

1 Introduction

Live fish recognition in the open sea has been investigated by [1, 2, 3, 4] to promote commercial and environmental applications like fish farming, meteorologic monitoring and fish quota monitoring. Computer vision and pattern recognition techniques can help biologists observe marine ecosystems where the manual annotation is too expensive, *i.e.* there are too many underwater videos (from a

tera-scale video database). The fish are swimming in general 3D freedom with a complex background including coral, sand and the open sea. Computer vision techniques can help detect significant events in such environments and filter out most worthless content from mass video databases. These techniques, when integrated with marine knowledge, can analyse underwater objects and compose high level interpretations, like fish counting, fish species distribution variation, fish behaviour analysis. Marine scientists benefit from computer-assisted analysis of underwater videos, *e.g.* fish detection and species recognition for long-term observation [5]. Statistics about specific oceanic fish species distributions or aggregate counts of aquatic animals can assist biologists with resolving issues ranging from food availability to predator-prey relationships [6, 7, 8]. However, fish recognition in open water is fundamentally challenging because fish can move freely and illumination levels change frequently in such environments [9, 10, 11]. As a result, this task remains an open research problem. Prior research is mainly restricted to constrained environments (*e.g.*, fish tanks [1], conveyor belts [12]). Strachan et al. [3] achieves a recognition rate of 73%, 63% and 90%, respectively, on three types of fish. Spampinato et al. [13] classifies 360 images of ten different species and achieves an average accuracy of about 92%. Larsen et al. [14] classify three fish species and achieve a recognition rate of 76%. In contrast, this paper investigates novel techniques to perform effective live fish recognition in an unrestricted natural environment.

The three main contributions of the paper are: (1) a novel hierarchical classification method suited for greatly unbalanced classes, (2) a novel classification-rejection method to clear up decisions and reject unknown classes, and (3) an application of the classification method to free swimming fish. A hierarchical classification method called the Balance-Enforced Optimized Tree with Reject option (BEOTR) method is introduced to process the fish samples from an imbalanced dataset. The reject option evaluates the *posterior* probability of input data and filters out less confident results, *e.g.* noise, classification errors or unknown classes. The framework, illustrated in Fig 1, includes feature extraction, hierarchical classification and rejection. Firstly, 2626 features are extracted after a fish orientation procedure. Then, a BEOTR of 15 fish species is automatically constructed using the inter-class similarities and a heuristic method. After a Forward Sequential Feature Selection (FSFS) and learning the finite mixture models, a GMM model is applied after the BEOTR method to evaluate the *posterior* probability of testing samples and provides the reject option.

The rest of the paper is organized as follows: Section 2 briefly introduces related work. Section 3 describes the proposed live fish recognition system with reject option. Section 4 shows experimental results in an underwater observational system and conclusions are drawn in Section 5.

2 Related work

Traditionally, marine biologists have employed many tools to examine the appearance and quantities of fish. For example, they cast nets to catch and recog-

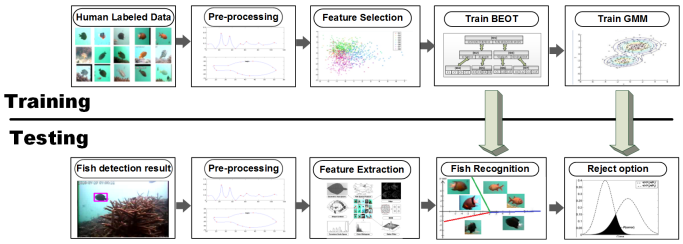


Figure 1: The framework of our BEOTR-based hierarchical classification system. The work flow shows the training and the recognition procedure. Section 3.1 introduces the pre-processing and feature extraction methods. Section 3.2 discusses the BEOTR hierarchical classification algorithm. The GMM method for reject option is described in Section 3.3.

nize fish in the ocean. They also dive to observe underwater environment, using photography [15]. Moreover, they combine net casting with acoustic (sonar) [16]. Nowadays, much more convenient tools are employed, such as hand-held video filming devices. Embedded video cameras are also used to record underwater animals (including insects, fish, *etc.*), and observe fish presence and habits at different times [17]. This equipment has produced large amounts of data, and it requires informatics technology like computer vision and pattern recognition to analyse and query the videos. Statistics about specific oceanic fish species distribution, besides an aggregate count of aquatic animals, can assist biologists resolving issues ranging from food availability to predator-prey relationships [6]. Unlike the simple and constrained environments found in the majority of previous work (*e.g.*, fish tanks [1, 2], conveyor belts [12], dead fish [14]), we investigate the recognition task of more fish species in a more complex and fundamentally challenging natural environment. We use underwater camera to record and recognize fish, where the fish can move freely and the illumination levels change frequently both locally from caustics arisen from the ocean surface waves and globally due to the sun and cloud positions [10].

2.1 Hierarchical classification method

The task of fish recognition is an application of multi-class classification, which has become an important and interesting research area since the influence of machine learning theory. Over the last decade, SVM [18] has shown impressive accuracy on the multi-class classification task because of its maximum-margin advantages.

Given a training set $\mathcal{D}$ from p classes, which is a set of n sample points of the form:

$$\mathcal{D} = \{(\mathbf{x}_i, y_i) \mid \mathbf{x}_i \in \mathbb{R}^m,\ y_i \in \{1, ..., p\}\}_{i=1}^{n} \tag{1}$$

y_i indicates the class label of m-dimensional vector $\mathbf{x}_i$. Considering the two-class task ($p = 2$), a Support Vector Machine (SVM) [19] is optimized to find a

hyperplane, called maximum-margin hyperplane, which maximizes the margin between the two classes.

SVMs were initially designed to be a binary classifier. They can be adapted to form a multiclass classifier by converting the single multiclass problem into multiple binary classification problems [20]. This kind of multi-class classifier could be considered as a flat classifier because it classifies all classes at the same time [21] and omits the inter-class correlations. A shortcoming of the flat classifier is that it uses the same features to classify all classes without considering that some classes have certain similarities and can be better separated by some customized features at a later stage. To overcome the problem of flat classifiers, one possible solution is to integrate a domain knowledge database with the flat classifier and construct a tree to organize all classes hierarchically [22]. This strategy is called hierarchical classification which inherits from the divide and conquer tactic. Essentially, it uses a hierarchical classification procedure where a customized classifier is trained with specific features at each level [23].

A taxonomy tree is a typical biological hierarchical classification method. The taxonomy ontology aims to systematize animals into their hierarchical categories. Taxon, as the leaf nodes of the whole tree, are the basement of taxonomy knowledge. For each taxon in the taxonomic tree, there is a top-to-bottom description to identify its hierarchical information which contains several concepts, known as Kingdom, Phylum, Class, Order, Family, Genus, Species. The taxonomy methodology is based on the synapomorphies characteristic from the extent to which the taxon is monophyletic, and it indicates the distinction between species, *e.g.* the presence or absence of components (anal-fin, nasal, infraorbitals), particular numbers (six dorsal-fin spines, two spiny dorsal-fins), particular shape (second dorsal-fin spine long, thick caniniform teeth), *etc.* We used the biological taxonomy knowledge to help construct a hierarchical classification tree of the 15 most common fish species as a baseline hierarchical classification method. This tree splits all classes into nine groups at the first level according to their family synapomorphies characteristic and leaves a few similar species to a deeper layer where a customized classifier is used.

Hierarchical classification has several noticeable advantages. Firstly, it divides all classes into certain subsets and leaves similar classes for a later stage. This strategy helps reduce the imbalance of data. Secondly, unlike the flat classifier choosing a feature set based on the average accuracy over all classes, the hierarchical method applies a customized set of features to classify specific classes. As a result, it achieves better performance on similar classes. Thirdly, the hierarchical solution exploits the correlations between classes and finds the similar groupings. This is especially useful with a large number of categories [22]. Hierarchical structures are popular in document and image categorization. Mathis [24] organizes documents hierarchically by making use of the correlations between topical subjects. Deng *et.al.* [25] introduced a new dataset called ImageNet where a large scale hierarchical ontology of images are constructed based on the WordNet knowledge. However, these approaches use pre-defined hierarchical structures without considering how to construct a more accurate tree based on specific classes and their properties.

2.2 Classification with reject option

We apply a pattern recognition method to recognize fish in underwater videos. The classification procedure is carried out by a hierarchical algorithm. One problem with these hierarchical classification methods is error accumulation. Each level of the hierarchical tree has some classification errors. These misclassified samples reduce the average accuracy of the final recognition performance. Another issue for a multi-class classifier (not only for hierarchical classification) is that it classifies every test sample into one of the training classes. Although our fish recognition dataset covers the 15 most dominant species of fish, there are still many observed fish from unmodeled species. These fish images are classified as known species, and the precision is thus decreased. A "reject option" for a multi-class classifier allows the classifier to reject less confident classification results as recognition errors or unknown classes. The goal is to detect whether the features are properly clustered within each class, and indicated strong class purity and consistency. If accumulated errors in the hierarchy misclassifications have occurred, then the samples are expected to have low likelihood scores. The reject function evaluates the posterior probability of the test samples given their recognition result. This is a post-recognition step and the rejection is independent from the recognition since it only conditions on the recognition result. Furthermore, manual annotation work for these minority species is expensive because of the small proportion of these images, when compared to the major species.

Thus, the reject option helps the fish recognition application in finding new species. It eliminates the samples that are dissimilar to the assigned classes. Thus, a p-class SVM has $p+1$ decisions: $\{1, ..., p, Reject\}$. The reject option means either a wrong decision of any of the p classes or the sample is from an unknown class.

Platt [26] proposed a rejection method that used an additional sigmoid function $P(y = 1 \mid t) = 1/(1 + \exp(at + b))$ to map the SVM outputs into *posterior* probabilities $P(y = \pm 1 \mid t)$ rather than first estimating the class-conditional probabilities $P(t \mid y = \pm 1)$, where t is the SVM output, a and b are parameters trained from the validation set. The *posterior* probability mapping function can be estimated by using the maximum conditional likelihood [27]:

$$< a, b >= \underset{a,b}{\operatorname{argmax}} P(y = 1 \mid t, a, b) = \underset{a,b}{\operatorname{argmax}} \prod_i P(y = 1 \mid t^i, a, b), \forall_i \quad (2)$$

where t^i denotes the output for the ith validation sample, and y is the class label.

Another common way to give a score to the classifier decisions is the Soft-Decision hierarchical classifier. In [27], Wang *et al.* present an implementation using the SVRDM classifier. The significant change is that there is no constraint that the output of each node should sum to one. Given evidence $\mathbf{X}$ and the classification result for each sub-branch m, each node i in the classification path

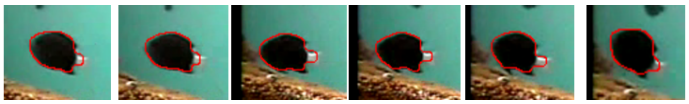


Figure 2: An example of fish detections from a whole trajectory of *Chromis margaritifer*. This species of fish has a noteworthy tail of white colour. This feature is essential to discriminate it from other species of fish. These figures have successfully maintained most of the white tails.

generates a probability output $P_i(C = m \mid \mathbf{X})$. The final *posterior* probability P is the product of the corresponding P_i along each path. However, these methods have difficulties in how to evaluate the certainty score based on the intermediate results at different layers of the hierarchical classification tree.

3 Methodology

In this section, we present our novel hierarchical classification method called Balance-Enforced Optimized Tree with Reject option (BEOTR). It improves the normal hierarchical method by arranging more accurate classifications at a higher level and keeping the hierarchical tree balanced. Since hierarchical methods accumulate errors along the decision path, a novel rejection system is provided as an alternative channel to discover misclassified samples and to probe test samples from new classes at the leaves of the classification hierarchy. The whole system consists of three stages: feature extraction, hierarchical classification and result rejection (illustrated in Figure 1). Section 3.1 introduced the pre-processing and feature extraction methods. Section 3.2 discusses the BEOTR hierarchical classification algorithm. The GMM method for reject option is described in Section 3.3.

3.1 Data pre-processing and feature extraction

Pre-processing is undertaken to improve the quality of features. Firstly, the detection and tracking software described in [17] is used to obtain the fish and mask images. Then the Grabcut algorithm [28] is employed to segment fish from the background. An example of the detected fish is in Figure 2 of *Chromis margaritifer* where most parts of the key feature (white tail) are preserved by the segmentation algorithm.

However, an issue here can be observed is that the detected contour (marked by red string along the body of fish) does not exactly align with the foreground outline. Pieces of the dorsal fin, besides a portion of the anal fin, are missing while some background such as water area is falsely included in front of the fish. These inaccurate segmentations give a distorted image of the fish outline and lose detailed descriptions when generating the shape features. Thus, we append texture and colour features besides the shape features to produce a

more comprehensive and robust set of descriptors. In order to align the fish images to the same direction before further processing, we propose a streamline hypothesis, which uses the assumption that most fish have a smoother head than tail because fish need a more frictional tail (caudal fin) to swim and help them keep balance. In order to find the tail side, we smooth the fish boundary with a Gaussian filter to eliminate some noise, and then calculate the curvature of each boundary pixel as following [29, 30]:

$$\kappa(u,\sigma) = \frac{X_u(u,\sigma)Y_{uu}(u,\sigma) - X_{uu}(u,\sigma)Y_u(u,\sigma)}{(X_u(u,\sigma)^2 + Y_u(u,\sigma)^2))^{\frac{3}{2}}} \tag{3}$$

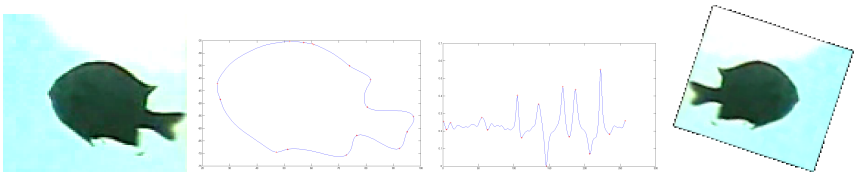
where $X_u(u,\sigma)/X_{uu}(u,\sigma)$ and $Y_u(u,\sigma)/Y_{uu}(u,\sigma)$ are the first and the second derivative of $X(u,\sigma)$ and $Y(u,\sigma)$, respectively; $X(u,\sigma)$ and $Y(u,\sigma)$ are the convolution result of 1-D Gaussian kernel function $g(u,\sigma)$ with fish boundary coordinates $x(u)$ and $y(u)$. We fix σ to 0.35, so that κ only depends on u. However, the pixel curvature is sensitive to local corners and we normalize it using the logarithm function:

$$\kappa_{normalize} = \begin{cases} log(\kappa) & \text{if } \kappa \geq 1 \\ -log(2-\kappa) & \text{if } \kappa < 1 \end{cases} \tag{4}$$

The fish boundary coordinates are weighted by their local curvature and the vector from the centre of mask to the curvature weighed centre estimates the tail orientation. We use this algorithm to align all fish horizontally where the head of the fish is located on the right. A typical fish orientation procedure is illustrated in Figure 3. Considering the first image (Figure 3(a)) as input, we first smooth the contour image with a Gaussian filter to eliminate the spines, which generates a pulse in curvature and should be excluded since we only care about substantial components (Figure 3(b)). The curvature of the fish contour is illustrated in Figure 3(c), where the X axis is the index of pixels of contour starting from the fish top part in anti-clockwise and the y axis stands for the curvature degree. The curvature degree severely fluctuates more on the right side than the left on the plot since the curvature concentrates on the rear half of fish. We apply an ellipse fitting for minor adjustment. It calculates the angle between the x-axis and the major axis of the ellipse that has the same second-moments as the fish mask. Figure 3(d) shows the final result, where the *Dascyllus reticulatus* is rotated horizontally and faces right. The fish orientation method achieves 95% correct fish orientation with ± 15 degrees based on 1000 manually labelled fish images. Finally, every fish image is divided into four parts (head/tail/top/bottom) by splitting the fish horizontally and vertically through the centroid.

Next, 66 types of feature are extracted. These features are a combination of colour, shape and texture properties in different parts of the fish such as tail, head, top, bottom and the whole fish.

The first introduced feature are normalized colour histograms in the Red&Green channel and the Hue component in HSV colour space. These colour features are



(a) . (b) . (c) . (d) .

Figure 3: Fish orientation demonstration: (a) original fish image; (b) fish boundary after Gaussian filter; (c) curvature along fish boundary; (d) oriented fish image.

normalized to minimize the effect of illumination changes. We recompute the range of every bin according to the average distribution over all samples and map them into an 11-bin histogram to take full advantage of all bins. This is done for every colour channel separately, as shown below:

$$\widetilde{B}_i = \sum_{j=a_i}^{a_{i+1}} B_j$$
$$s.t. \qquad a_i = min\{X \in \mathbb{N}^+ \mid \Sigma_{j=1}^{X} \overline{B}_j \geq \frac{i}{11}\} \quad (5)$$

where $B_j, j \in \{1, ..., 50\}$ is the original colour histogram bin, $\overline{B}_j, j \in \{1, ..., 50\}$ is the averaged histogram over all samples and $\widetilde{B}_i, i \in \{1, ..., 11\}$ is the recomputed bin.

In order to describe the fish texture, we calculate the co-occurrence matrix, Fourier descriptor and Gabor filters. The grey level co-occurrence matrices describe the co-occurrence frequency of two grey scale pixels at a given distance d [13]:

$$C_{\Delta u, \Delta v}(i,j) = \sum_{p=1}^{n} \sum_{q=1}^{m} \begin{cases} 1, & \text{if } I(p,q) = i \text{ and} \\ & I(p+\Delta u, q+\Delta v) = j \\ 0, & \text{otherwise} \end{cases} \quad (6)$$

The frequency is calculated for 4 orientations λ. We compute Contrast, Correlation, Energy, Entropy, Homogeneity, Variance, Inverse Difference Moment, Cluster Shade, Cluster Prominence, Max Probability, Auto correlation, Dissimilarity. These 12 features are useful as they are the most commonly selected features by the Forward Sequential Feature Selection (FSFS) procedure [31].

Moment Invariants [32] and Affine Moment Invariants [33], as well as Pyramid of Histograms of Orientation Gradients (PHOG) [34], are employed as the shape features. Furthermore, some specific features like tail/head area ratio, tail/body area ratio, *etc.* are also included. All features are normalized by subtracting the mean and dividing by the standard deviation (z-score normalized after 5% outlier removal).

3.2 Algorithm for constructing the hierarchical classification tree

Given a set of samples $\{\mathbf{x}_i\}_{i=1}^n$, the feature vector $\mathbf{f}_i = \{f_{i,1}, ..., f_{i,m}\}$ denotes the m feature values for sample $\mathbf{x}_i$. Let $\{y_i\}_{i=1}^n$ indicate the class label of $\mathbf{x}_i$, and $y_i \in \{1, ..., C\}$ where C is the number of classes. Our aim is to construct a classifier h which uses the feature $\mathbf{f}_i$ as input to predict the class label $\tilde{\mathrm{y}}_i = h(\mathbf{f}_i)$ that maximizes the classification accuracy.

A hierarchical classifier h_{hier} is designed as a structured node set. A node is defined as a triple: $\mathrm{Node}_t = \{\mathrm{ID}_t, \tilde{\mathrm{F}}_t, \hat{\mathrm{C}}_t\}$, where ID_t is a unique node number, $\tilde{\mathrm{F}}_t \subset \{\mathbf{f}_1, ..., \mathbf{f}_m\}$ is a feature subset chosen by a Forward Sequential Feature Selection procedure that is found to be effective for classifying $\hat{\mathrm{C}}_t$, which is a subset of classes and their groups. We only consider binary splits so each node has at most two groups. All samples that are classified as the same group will be transmitted into the same child node for later processing. An example with 15 classes is shown in Figure 9, where the ID_t is illustrated in each node and $\hat{\mathrm{C}}_t$ are the local groups. The binary splitting process stops when each group has at most 4 classes (*e.g.* Node ID 4,5,6,7) in order to limit the maximum depth of the tree and avoid overtraining. All the leaf nodes are multiclass SVMs using One-versus-One strategy.

To construct a hierarchical tree, we first aim at finding an optimal split of the given classes at the current node by minimizing the Average Recall (AR) score between the two child nodes. We search all possible splits of the classes into two nearly equal sets of classes. This process is repeated for each child node. A well-designed hierarchical tree can help improve the accuracy of some confusable classes while suppressing the error accumulation. In this paper, we propose two heuristics for how to organize a single classifier and construct a hierarchical tree with higher accuracy.

1. Arrange more accurate classifications at a higher level and leave similar classes to deeper layers.

2. Keep the hierarchical tree balanced to minimize the max-depth and control error accumulation (Here we split the tree by equal number of classes, but one could also use other splits, such as by equal *a prior* probabilities, or non-equal numbers of classes to minimizing error).

When constructing the hierarchical tree, we focus on balanced trees for computational reasons, and because a balanced tree structure produces reduced tree depth, which reduces error accumulation. More formally, our tree generation algorithm can be described as follows:

Input: class C_1 to C_n
begin
$c := \{C_1, ..., C_n\}$

```
  level := 0
  // Use Forward Sequential Feature Selection
  featureSetAllClasses := FeatureSelection(c)
  construct(c, level)
end
proc construct(c, n) ≡
  if n > MAXDEPTH
    exit
  end
  // Evaluate classification accuracy on each
  // split of classes c in parallel
  parallel for {binary splits of c}
      r = evaluate(c, AllFeature)
  end
  // The ChooseSplit finds the optimal class
  // subset pair based on the set of r evaluations
  [cLeft, cRight] := ChooseSplit({r})
  // Use Forward Sequential Feature Selection
  featureSet := FeatureSelection(cLeft, cRight)
  // The maximum leaf node subset
  // size is set to 4 to limit max tree depth
  if size(|cLeft|) > 4
    construct(cLeft, n + 1)
  end
  if size(|cRight|) > 4
    construct(cRight, n + 1)
  end
end
```

Firstly, the algorithm splits the current set of classes c into all combinations of pairs of disjoint subsets with size $\frac{|c|}{2}$ and then sends each combination to the performance evaluation stage. This consists of training using the training subset and feature selection using the validation set. Training uses Forward Sequential Feature Selection to build a binary SVM classifier. After evaluating all of the possible splits, the best subset pair, in terms of classification accuracy, is chosen and this split is used to construct two new child tree nodes. This procedure is iterated for both child branches until the stopping criteria is satisfied. Performance evaluation of each subset at a given tree level is independent of every other split. We assign each combination of class set splits to a distributed parallel task. Each pair of subsets is then evaluated to obtain an accuracy score in parallel (the accuracy score for each distributed task is found by taking the mean classification accuracy of the two subsets assigned to the task). After all distributed tasks in a superstep have concluded, we collect all of the mean accuracy scores and select the class split with the highest score (our superstep conclusion).

An example of an automatically generated tree is shown in Fig 9, where

15 classes are arranged into 3 layers. The first layer splits all classes into two groups: C1, C2, C3, C4, C7, C8, C9, C11 and C5, C6, C10, C12, C13, C14, C15. Then it chooses the feature subset to maximize the average accuracy of these groups. This procedure keeps on until all groups have at most 4 classes.

3.3 Gaussian mixture model for reject option

In a p-class classification application, an input sample is predicted as one of the p classes. This solution cannot process noisy data *e.g.* false detections, samples from unknown classes. These noisy data should be identified and then rejected. We thus apply a Gaussian Mixture Model (GMM) to evaluate the posterior probability from known classes to reject unreliable results. A GMM is a semi-parametric density model which is comprised by a number of Gaussian components [35]. It assumes that the data features are originally sampled from a weighted sum of multiple Gaussian functions. In feature space, a GMM provides more flexibility and precision in modelling the underlying statistics of sample data [36].

A GMM represents the hypothetical clusters of density distributions in feature space when a single component Gaussian cannot model the underlying characteristics of the given class. For example, in fish recognition, some species of fish have specific colours, fin shapes, stripes or texture. It is reasonable to assume that the extracted features represent this domain knowledge and represent them by the density distributions. Each characteristic is expressed both by the mean value μ_i and the covariance matrix Σ_i. The training procedure is unsupervised (after assigned the training class). The GMM captures the prominent density distributions and is not constrained by the label information. The Expectation Maximization (EM) algorithm [37], which is guaranteed to converge to a local maximum by iterative searching, is applied to optimize the Gaussian mixture model. Figueiredo *et al.* [38] present an unsupervised learning algorithm to learn a proper mixture model from multivariate data. We use it to select the number of components in the finite mixture model. Their approach uses the Minimum Message Length (MML) with advantages compared to other deterministic criterion, *e.g.* BIC [39], MDL [40], *etc*: less sensitive to the initialization, avoids the boundary of the parameter space.

A GMM model is applied after the hierarchical classification (at the leaves of the hierarchical tree) to implement the reject option, instead of calculating the result score along the path in a hierarchy (Figure 9). One difficulty for rejection in a hierarchical method is how to evaluate the certainty score based on the intermediate classification results at different layers. Instead of integrating the result score along the path of the hierarchy [27], here a GMM model is applied after the BEOTR classification to implement the reject option (Figure 9). The GMM model is estimated using the training data for a subset of features selected using the Forward Sequential Feature Selection method [31]. For each BEOTR result, the final *posterior* probability $P(C \mid \mathbf{x})$ for that input is evaluated to estimate the final probability that the sample is actually class C output according to the GMM likelihood score. Samples with a low probability are

rejected. Thus, a lower false positive rate is expected, since some misclassified samples in the BEOTR classifier can be rejected at the price of a slightly lower true positive rate due to incorrect rejection. Rejected samples are labelled as unrecognized. Note that in Figure 9 the reject option is only used for classes $C1 - C6$ because classes $C7 - C15$ had too few samples to properly train the GMM after Forward Sequential Feature Selection. Rejection is based on the *posterior* probability for the observation evidence X with the predicted class C_i (using Bayes' Rule):

$$p(C_i \mid X) = \frac{p(C_i)p(X \mid C_i)}{p(X)} = \frac{p(C_i)p(X \mid C_i)}{\sum_j p(C_j)p(X \mid C_j)} \tag{7}$$

The prior knowledge $p(C_i)$ is calculated from the training dataset, $p(X \mid C_i)$ is the probability of the evidence given the predicted class C_i. Figure 4 illustrates the distribution of the *posterior* probability $p(C_i \mid X)$ of all samples that are classified as species *Chromis chrysura*. These samples are either correctly classified (True Positives) or misclassified (False Positives). The distribution of *posterior* probability of False Positives (as shown in Figure 4 c) has a peak distribution (about 38%) around the value of zero while most of the True Positives have higher *posterior* probability (Figure 4 b). The diversity between these two distributions is exploited to distinguish False Positives. This algorithm rejects a substantial portion of the misclassified samples with the cost of also rejecting a small proportion of True Positives (see Section 4 for details).

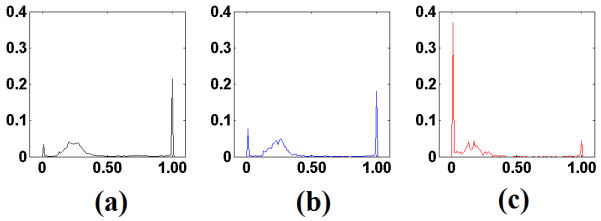


Figure 4: (a) Distribution of *posterior* probability of the training samples of species *Chromis chrysura*. (b) Distribution of *posterior* probability of True Positives. (c) Distribution of *posterior* probability of False Positives. See text for details.

4 Experiment with fish recognition

Our data is acquired from a live fish dataset of the 15 different species shown in Figure 5. This figure shows the fish species name and the numbers of ground-truth images. The data is very imbalanced where the most frequent species is about 500 times more common than the least common one. Note, the images shown here are ideal images as many of the others in the database are a bit

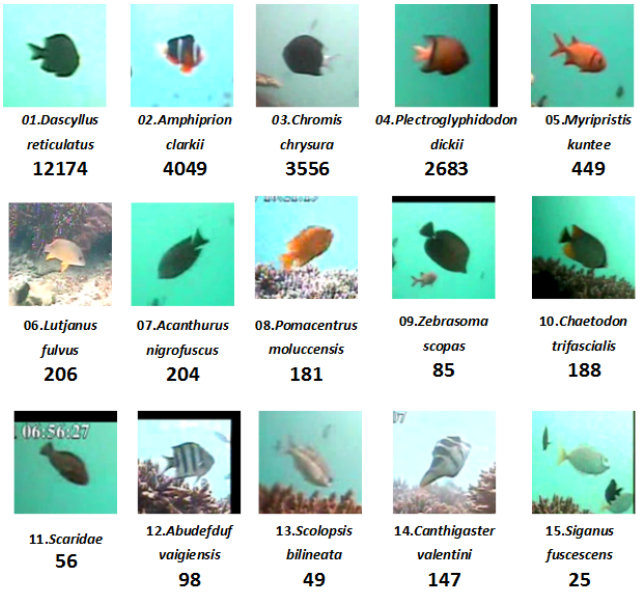


Figure 5: Top 15 species of fish in underwater videos, with the number of observations in the ground-truth.

Figure 6: Hard fish examples, due to blurry conditions, different distances/orientations, against coral or ocean floor backgrounds.

blurry, and have fish at different distances and orientations or are against coral or ocean floor backgrounds. Figure 6 shows some hard fish examples.

All fish are manually labelled by following instructions from marine biologists [41]. 2626 dimensions of feature are extracted as described in Section 3.1.

4.1 Hierarchical classification for fish recognition

We use the BEOTR method for fish recognition. Based on the multi-class classifier, we designed five other classifiers:

1. A multiclass 1v1 flat SVM classifier, which classifies all 15 classes simultaneously, is implemented as a baseline flat SVM classifier. Forward Sequential Feature Selection (FSFS) is applied (named flatSVM-fs) to do greedy selection of the features to maximize the average recall among all classes.

2. The Principal Component Analysis (PCA) algorithm is also implemented as a baseline method for feature selection and classification. It uses the singular value decomposition (SVD) to reduce the feature dimensions and we preserve 98% of the principal component variances (up to 583 dimensions). The processed features are then classified by a 15-class SVM classifier.

3. A classical classification and regression tree method (CART [42]) is provided as another automatically generated hierarchical decision tree to be compared with. It starts with a single node, and then looks for the binary distinction which gives the most information about the class. The generating process continues until it reaches the stopping criterion.

4. A taxonomy tree is constructed according to the fish species taxonomy. This tree is pre-defined. It reflects the homologous similarity between species. All the 15 species of fish belong to the Actinopterygii class (ray-finned fishes), but in different orders, families and genus. This tree splits all classes into 5 groups at the first level according to their family synapomorphies characteristic and leaves a few similar species to a deeper layers where the customized multiclass 1v1 SVM classifier is applicable.

5. An automatically generated tree (BEOTR) is designed by recursively choosing a binary split which has the best accuracy over the given classes. We choose binary splitting to keep the tree balanced.

The experiment is based on 24150 fish images with a 5-fold cross validation procedure with leave-one-out strategy. The training (3/5's) and testing sets (1/5's) are isolated so fish images from the same trajectory sequence are not used during both training and testing.

To make use of the temporal information, we developed a trajectory voting algorithm. Trajectory voting in Figure 7 is used to minimize the environmental influence. As all fish are freely swimming in a varying illumination environment, the detected fish may have different orientations and appearances. Therefore, the recognition results may vary even for a fish in the same trajectory. A trajectory based winner-take-all voting mechanism is applied after individual classification. It combines the single frame classification results. The trajectory voting method enhances the fish recognition accuracy by exploiting the consistency in labels expected from tracking each fish individually.

Results for the 5 algorithms are listed in Table 1 where the AR and AP are recall/precision averaged over all classes rather than over all fish. This is because of the greatly unbalanced class sizes. Three performance metrics are employed to evaluate the accuracy of the proposed system. The first metric is Average Recall (AR) over all species. It describes on average how many fish are correctly recognized for each species. This score is more important to our experiment because of the imbalance in the classes. Given True Positive / False

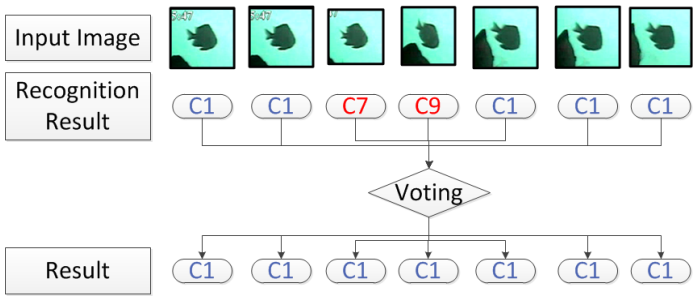


Figure 7: An example of trajectory voting is shown where we use a winner-take-all strategy.

Positive / False Negative, AR is defined as:

$$AR = \frac{1}{c}\sum_{j=1}^{c}\left(\frac{TruePositive_j}{TruePositive_j + FalseNegative_j}\right) \tag{8}$$

where c is the number of classes. The second score is Average Precision (AP) over all species. It is the probability that the classification results are relevant to specified species:

$$AP = \frac{1}{c}\sum_{j=1}^{c}\left(\frac{TruePositive_j}{TruePositive_j + FalsePositive_j}\right) \tag{9}$$

The third metric is the accuracy over all samples (Accuracy over Count, AC), which is defined as the proportion of correct classified samples among the whole dataset. AC is calculated as:

$$AC = \frac{\sum_{j=1}^{c} TruePositive_j}{\sum_{j=1}^{c}(TruePositive_j + FalsePositive_j)} \tag{10}$$

We compare the hierarchical classification against the flat SVM classifier (AR = 76.9%). PCA is a popular algorithm to reduce feature dimensions. We apply it before an SVM and achieve almost the same score (AR = 77.7%). In the third row, feature selection before use in a SVM produces slightly better results (AR = 78.4%) than the flat SVM using all features. Furthermore, the CART algorithm gets the lowest AR (53.6%) among all three hierarchical methods. The taxonomy methodology achieves a better AR of 76.1% than the CART but is worse than the automatically generated hierarchical tree (84.8%) which chooses the best splitting by exhaustively searching all possible combinations while remaining balanced. Most algorithms achieve high AC scores, but this is because the classes are very unbalanced. For example, to simply label all fish as class 1 already achieves an AC = 50.4%.

Method	AR (%)	AP (%)	AC (%)
flatSVM	76.9 ± 4.6	88.5 ± 3.6	95.7 ± 0.5
PCA (98%)	77.7 ± 3.8	88.9 ± 4.1	95.4 ± 0.4
flatSVM-fs	78.4 ± 3.7	88.0 ± 5.5	95.9 ± 0.4
CART [42]	53.6 ± 5.1	52.9 ± 4.6	87.0 ± 0.7
Taxonomy	76.1 ± 5.2	87.2 ± 6.7	95.3 ± 0.4
BEOTR	**84.8*** ± 3.9	**91.4** ± 2.8	**97.5*** ± 0.6

Table 1: Fish recognition results (before rejection). We add the standard deviation of AR/AP/AC over 5-fold cross validation. * means the score is a significant improvement after t-test over other methods at 95% confidence level.

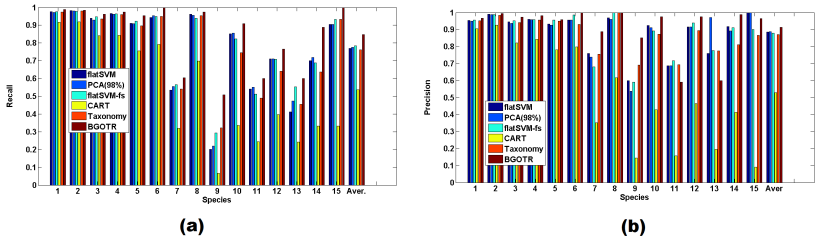


Figure 8: Recall (a) and precision (b) of 15 species. These scores are averaged by 5-fold cross validation.

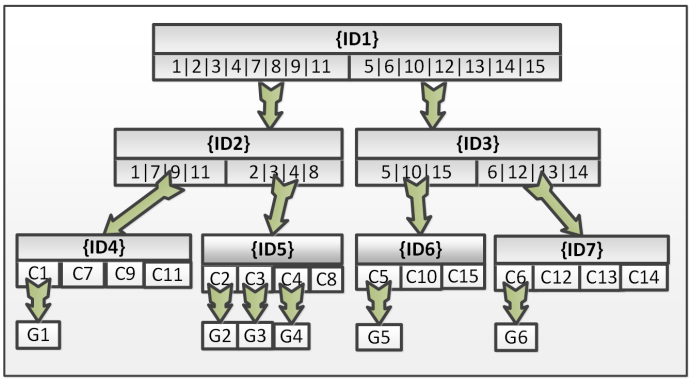


Figure 9: Automatically generated tree, the hierarchical example tree of 15 classes ($C1, ..., C_{15}$). GMMs (G1,...,G6) for the rejection in hierarchical classification is integrated with the BEOTR method.

The individual class recalls/precisions are shown in Figure 8. The hierarchical approaches achieve better accuracy than the flat SVM classifier and other baseline methods because they arrange the similar species into the same group and add fish-tail features to distinguish these species. Species 7,9,11,13 have low scores in part due to confusion with the much larger class 1.

4.2 Fish recognition with rejection

We evaluate the reject option with an application of fish recognition. The experiment is carried out by comparing our GMM-based method with two other state-of-the-art rejection methods: 1) relating SVM outputs to probabilities [26], and 2) soft-decision hierarchical classification with a reject option [27]. We provide a reject option for the top 6 species (23117 fish images) in the top 6 species of Figure 5. In order to test the performance in probing unknown classes, another 3220 fish images are added as the new species, where none of them is from the top 15 species. These samples are considered as false classes from the detection step and they are supposed to be rejected. Note that the BEOTR tree has no "none of the above" classification. Hence any fish that did not belong to one of the 15 trained species would be classified as one of the 15. The reject option allows some of the unknown fish to be rejected.

For each fish species, we trained a GMM with the selected feature subset by Forward Sequential Feature Selection method. The features used for training the GMM are the same as for training the hierarchical tree but a different subset was selected. The average numbers of features selected across all folds were: *D. reticulatus*: 67, *A. clarkii*: 42, *C. chrysura*: 66, *P. dickii*: 26, *M. kuntee*: 46, *L. fulvus*: 7. We used [38] to select the number of mixture models where the maximum number of Gaussian density components is set as 7. In [43], Chib

Species	TR (known classes)		TR (unknown classes)	
	rate(%)	number	rate(%)	number
D. reticulatus	13.7	15	11.2	33
A. clarkii	20.3	4	11.4	212
C. chrysura	32.8	15	51.2	53
P. dickii	13.9	6	14.8	19
M. kuntee	41.7	6	80.6	13
L. fulvus	65.7	4	48.6	106
Total	20.9	50	16.7	436

Table 2: Rejection results of incorrect classifications from either the trained 15 species (cols 2,3) or new 8 species (cols 4,5). (TR=True Rejection). Last row is the averaged score weighted by size of each set. On average, there were 247 incorrectly classified samples from the 15 known classes and 2627 samples from unknown classes.

Species	True positives		False Rejection	
	rate(%)	number	rate(%)	number
D. reticulatus	91.9	2237	4.1	95
A. clarkii	95.7	775	0.7	6
C. chrysura	85.2	606	8.0	53
P. dickii	92.5	496	1.8	9
M. kuntee	80.4	74	2.1	1
L. fulvus	84.2	35	1.7	1
Total	91.3	4223	3.7	165

Table 3: True positive rate when using only the 15 class data after rejection (cols 2,3) and additional false rejections due to the rejection step (cols 4,5). Last row is the averaged score weighted by size of each set.

and Siddhartha express the marginal density as the prior probability times the likelihood function over the *posterior* density. They found comparably improved performance of the marginal likelihood with this estimation. Since we address the improvement of rejection in the hierarchical classification, we also calculate the *posterior* density from Bayes rule of the testing samples. More specifically, for each sample with evidence X and BEOTR prediction $\tilde{C}$, we calculate its *posterior* probability $P(C \mid X)$ from Equation 7, and set a small threshold (*i.e.* 0.01) to reject all samples whose *posterior* probabilities are below the threshold (Figure 4).

Tables 2 and 3 demonstrate that GMM effectively improves the reject option in the hierarchical classification for fish recognition. Table 2 shows the incorrect classification rejection results. The second/third columns show how many misclassified samples from the top 15 species are correctly rejected while the

fourth/fifth columns show correctly rejected samples from the new species. In table 3, columns 2 and 3 show the true positive rate when using only the 15 class data after rejection and the last two columns show how many correctly classified fish are thrown out when we apply the reject option (False rejection rate). In a preferable situation, for samples classified as *e.g. Lutjanus fulvus*, 65.7% misclassified samples and 48.6% new species samples are rejected while only 1.7% of the correctly classified samples are falsely rejected. However, as fish can move freely and illumination levels change frequently in such environments, there are always some new testing samples belonging to the known species but their feature distributions in feature space are not effectively captured by the GMM. We have to balance the tradeoff between more rejection and more remaining. For example, the cost of the reject option for *Chromis chrysura* is that we throw away 8.0% (53 images) of correct fish while we have correctly rejected 32.8% and 51.2% of the wrongly classified fish from 15 species and new species, respectively.

Algorithm	AP (%)	AR (%)
BEOTR baseline (no rejection) [44]	56.5	**91.1**
BEOTR+SVM probabilities [26]	59.0	90.9
BEOTR+soft-decision hierarchy [27]	59.0	90.7
BEOTR+GMM (proposed method)	**65.0***	88.3

Table 4: Fish recognition result, averaged by the top 6 species. * means significant improvement with 95% confidence after t-test. None of the AR scores is significantly worse than the original BEOTR method. Note, the results in this table include both the 15 known species and the unknown species, hence the lower AP scores.

The experiment results in Table 4 demonstrate that our method rejects substantial numbers of misclassified samples as shown in Table 2 (20.9% and 16.7% True Rejection rates for known and unknown species, respectively) while the cost is that a small proportion of correctly classified samples are also rejected (3.7% False Rejection rate in Table 3). We compare it to two other rejection algorithms [26, 27] and our method achieves significant improvements in AP. Note that the AP in Table 4 is substantially worse than that in Table 1. This is because the experiment in Table 4 also included the 3220 fish of unknown species. The proposed method improves BEOTR hierarchical classification in two aspects: 1) filters out part of the misclassified samples and increase the average precision / count accuracy with a small reduction of the average recall, 2) finds potential new samples which do not belong to any known classes. It detects a set of samples which have a higher probability of coming from new species, and therefore, reduces the work for finding new fish, especially in a large database of underwater videos. To summarize our result, we use the F-score to combine both the average recall and the average precision of the test. We use the F_1 measure, which is the harmonic mean of precision and recall, as

Figure 10: Invalid fish images, chosen from 3 underwater videos. In a normal classifier without reject option, these images would be classified and cause unexpected results. Our BEOTR aims at eliminating them while preserving most valid fish images.

shown in Table 5. Considering both averaged recall and precision, our proposed method (BEOTR+GMM) achieves significant improvement compared to other three methods.

Algorithm	F_1-score
BEOTR baseline (no rejection) [44]	0.7135 ± 0.0227
BEOTR+SVM probabilities [26]	0.7150 ± 0.0222
BEOTR+soft-decision hierarchy [27]	0.7140 ± 0.0225
BEOTR+GMM (proposed method)	**0.7485 ± 0.0194** *

Table 5: F-score result of the top 6 species. * means significant improvement with 95% confidence after t-test.

4.3 BEOTR application to new real videos

Our fish recognition system depends on the detection results. Due to the complex environment (*e.g.* light distortion, fish occlusions and illumination transformation), the detection algorithm produces errors that are input to the classification procedure and cause unexpected recognition results. The previous experiments are evaluated on a "clear" dataset where all tested images are valid fish from either known or unknown species. However, in real applications, the acquired data may contain false detections, *e.g.* blurred images, occlusion by other fish or background objects, non-fish objects (coral, sea flowers, *etc.*). Some examples of false detections are shown in Figure 10. In this section we experimentally evaluate how many false detections our BEOTR system can reject while preserving the valid ones. We choose 3 underwater videos and have labelled 1000 detections from each video.

The recognition results are shown in Tables 6 and 7. We use BEOTR to classify the test images and calculate the Average Recall (AR) and Averaged Precision (AP) among all species. The AR score demonstrates that the BEOTR method recognizes about 78% of the real, untrained valid fish images correctly. The test images include many invalid detections (692, 892, 487, respectively). The BEOTR method filters more than half of these false detections (378, 705,

ID	Average Recall (AR)	Averaged Precision (AP)
video1	0.815	0.412
video2	0.804	0.448
video3	0.725	0.557
average	0.781	0.472

Table 6: Experiment results for real videos. In each video we select the first 1000 detections and manually label all samples.

ID	False detections	Rejections	TR	FR
video1	692	390	378	12
video2	852	734	705	29
video3	487	380	312	60
average	677	501	465	34

Table 7: Experiment of rejection results in real videos. TR = True Rejection, FR = False Rejection.

312, respectively) while it retains most of the valid inputs. Some false detections are not rejected and these inputs lower the average precision score (*c.* 47%).

5 Conclusion

In this paper, we present a novel Balance-Enforced Optimized Tree with Reject option (BEOTR) classifier for live fish recognition. This is a novel type of hierarchical classifier. More specifically, we propose a set of heuristics which are helpful to construct a hierarchical tree. A novel GMM-based rejection system is added after the recognition stage. We use feature selection to select a subset of features that best distinguishes samples of a given class from others. The reject function evaluates the *posterior* probability of the test samples and produces a lower false positive rate, but some misclassification errors in the BEOTR classifier can be overcome at the price of a slightly lower true positive rate due to incorrect rejection. The experimental results demonstrate the benefits of the rejection option, which results in an improvement in eliminating the accumulated errors from hierarchical classification compared to two other rejection algorithms. The proposed method is evaluated on a live fish dataset. This dataset of 24k samples over 15 species is the largest and most varied dataset used for fish species recognition. The automatically generated hierarchical tree with Reject option achieves *c.* 6% improvement of the average recall (AR) and *c.* 3% improvement of the average precision (AP) compared to the flat SVM and other hierarchical classifiers (Table 1). In the future, the hierarchical classification method could benefit from speeding up the construction procedure. Instead of choosing the best split by exhaustively searching all of the possible

combinations, a possible way is to exploit the pre-defined taxonomic hierarchical structure, which describes the hierarchical structure of fish species. This structure does not need any splitting calculation, and saves computing time. Here, we only explored the use of binary splits at the higher levels of the tree and terminated branching at node with 4 or less classes. Future work could explore other branching and termination criteria.

References

[1] D. Lee, R. B. Schoenberger, D. Shiozawa, X. Q. Xu, and P. C. Zhan, "Contour matching for a fish recognition and migration-monitoring system," *Proc. of SPIE*, vol. 5606, no. 1, pp. 37–48, 2004.

[2] B. P. Ruff, J. A. Marchant, and A. R. Frost, "Fish sizing and monitoring using a stereo image analysis system applied to fish farming," *Aquacultural engineering*, vol. 14, no. 2, p. 155–173, 1995.

[3] N. J. C. Strachan, P. Nesvadba, and A. R. Allen, "Fish species recognition by shape analysis of images," *Pattern Recognition*, vol. 23, no. 5, pp. 539–544, 1990.

[4] M. Okamoto, S. Morita, and T. Sato, "Fundamental study to estimate fish biomass around coral reef using 3-dimensional underwater video system," in *OCEANS 2000 MTS/IEEE Conference and Exhibition*, vol. 2, p. 1389–1392, 2000.

[5] D. Walther, D. R. Edgington, and C. Koch, "Detection and tracking of objects in underwater video," in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, vol. 1, pp. 544–549, 2004.

[6] A. Rova, G. Mori, and L. M. Dill, "One fish, two fish, butterfish, trumpeter: Recognizing fish in underwater video," in *IAPR Conference on Machine Vision Applications*, pp. 404–407, 2007.

[7] B. Zion, A. Shklyar, and I. Karplus, "In-vivo fish sorting by computer vision," *Aquacultural Engineering*, vol. 22, pp. 165–179, June 2000.

[8] M. R. Heithaus and L. M. Dill, "Food availability and tiger shark predation risk influence bottlenose dolphin habitat use," *Ecology*, vol. 83, no. 2, pp. 480–491, 2002.

[9] N. J. C. Strachan, "Length measurement of fish by computer vision," *Computers and electronics in agriculture*, vol. 8, no. 2, p. 93–104, 1993.

[10] Y. H. Toh, T. M. Ng, and B. K. Liew, "Automated fish counting using image processing," in *International Conference on Computational Intelligence and Software Engineering*, pp. 1–5, 2009.

[11] R. Schettini and S. Corchs, "Underwater image processing: state of the art of restoration and image enhancement methods," *EURASIP J. Adv. Signal Process*, vol. 2010, pp. 1–7, Jan. 2010.

[12] N. J. C. Strachan, "Recognition of fish species by colour and shape," *Image and Vision Computing*, vol. 11, pp. 2–10, Jan. 1993.

[13] C. Spampinato, D. Giordano, R. D. Salvo, Y. H. Chen-Burger, R. B. Fisher, and G. Nadarajan, "Automatic fish classification for underwater species behavior understanding," in *Proceedings of the first ACM international workshop on analysis and retrieval of tracked events and motion in imagery streams*, p. 45–50, 2010.

[14] R. Larsen, H. Ólafsdóttir, and B. Ersbøll, "Shape and texture based classification of fish species," in *Proceedings of SCIA*, p. 745–749, 2009.

[15] M. J. Caley, M. H. Carr, M. A. Hixon, T. P. Hughes, G. P. Jones, and B. A. Menge, "Recruitment and the local dynamics of open marine populations," *Annual Review of Ecology and Systematics*, vol. 27, pp. 477–500, Jan. 1996.

[16] P. Brehmer, T. D. Chi, and D. Mouillot, "Amphidromous fish school migration revealed by combining fixed sonar monitoring (horizontal beaming) with fishing data," *Journal of Experimental Marine Biology and Ecology*, vol. 334, pp. 139–150, June 2006.

[17] G. Nadarajan, Y. Chen-Burger, R. Fisher, and C. Spampinato, "A flexible system for automated composition of intelligent video analysis," in *Proc. of ISPA*, pp. 259–264, 2011.

[18] C. Chih-Chung and L. Chih-Jen, "LIBSVM: a library for support vector machines," *ACM Trans. Intell. Syst. Technol.*, vol. 2, no. 3, pp. 1–27:, 2011.

[19] C. Cortes and V. Vapnik, "Support-vector networks," *Mach. Learn.*, vol. 20, pp. 273–297, Sept. 1995.

[20] K.-B. Duan and S. S. Keerthi, "Which is the best multiclass SVM method? an empirical study," in *Proceedings of the 6th international conference on Multiple Classifier Systems*, MCS'05, pp. 278–285, Springer-Verlag, 2005.

[21] S. Carlos and F. Alex, "A survey of hierarchical classification across different application domains," *Data Mining and Knowledge Discovery*, vol. 22, no. 1-2, pp. 31–72, 2010.

[22] J. Deng, A. Berg, K. Li, and L. Fei-Fei, "What does classifying more than 10,000 image categories tell us?," in *ECCV*, vol. 6315, pp. 71–84, 2010.

[23] A. D. Gordon, "A review of hierarchical classification," *J. Royal Stat. Soc.*, vol. 150, no. 2, pp. 119–137, 1987.

[24] C. Mathis, "Classification using a hierarchical bayesian approach," in *Proc. of ICPR*, vol. 4, pp. 103–106, 2002.

[25] J. Deng, W. Dong, R. Socher, L. Li, K. Li, and L. Fei-Fei, "ImageNet: a large-scale hierarchical image database," in *CVPR*, pp. 248 –255, 2009.

[26] J. C. Platt, "Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods," in *Advances In Large Margin Classifiers*, pp. 61–74, MIT Press, 1999.

[27] Y.-C. F. Wang and D. Casasent, "A support vector hierarchical method for multi-class classification and rejection," in *International Joint Conference on Neural Networks, 2009. IJCNN 2009*, pp. 3281–3288, 2009.

[28] C. Rother, V. Kolmogorov, and A. Blake, "Grabcut - interactive foreground extraction using iterated graph cuts," *ACM Transactions on Graphics (SIGGRAPH)*, August 2004.

[29] F. Mokhtarian and R. Suomela, "Robust image corner detection through curvature scale space," *IEEE Transactions on PAMI*, vol. 20, no. 12, pp. 1376–1381, 1998.

[30] X. C. He and N. H. C. Yung, "Curvature scale space corner detector with adaptive threshold and dynamic region of support," in *Pattern Recognition, International Conference on*, vol. 2, pp. 791–794, IEEE Computer Society, 2004.

[31] Y. Saeys, I. n. Inza, and P. Larrañaga, "A review of feature selection techniques in bioinformatics," *Bioinformatics*, vol. 23, no. 19, pp. 2507–2517, 2007.

[32] J. Flusser, B. Zitova, and T. Suk, *Moments and moment invariants in pattern recognition*. John Wiley & Sons, 2009.

[33] J. Flusser, T. Suk, and B. Zitov, "Affine moment invariants," in *Moments and Moment Invariants in Pattern Recognition*, p. 49C112, John Wiley & Sons, Ltd, 2009.

[34] A. Bosch, A. Zisserman, and X. Munoz, "Representing shape with a spatial pyramid kernel," in *Proceedings of the 6th ACM International Conference on Image and Video Retrieval*, CIVR '07, (New York, NY, USA), p. 401C408, ACM, 2007.

[35] C. M. Bishop, *Neural Networks for Pattern Recognition*. Oxford University Press, Inc., 1995.

[36] S. J. Mckenna, S. Gong, and Y. Raja, "Modelling facial colour and identity with gaussian mixtures," *Pattern Recognition*, vol. 31, pp. 1883–1892, Dec. 1998.

[37] N. Shental, A. Bar-hillel, T. Hertz, and D. Weinshall, "Computing gaussian mixture models with EM using equivalence constraints," in *Advances in Neural Information Processing Systems 16*, MIT Press, 2003.

[38] M. A. T. Figueiredo and A. Jain, "Unsupervised learning of finite mixture models," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 24, no. 3, pp. 381–396, 2002.

[39] Y. Zhao, G. Karypis, and D.-Z. Du, *Criterion functions for document clustering*. PhD thesis, University of Minnesota, 2005.

[40] J. Rissanen, "Stochastic complexity and modeling," *The Annals of Statistics*, pp. 1080–1100, 1986.

[41] B. Boom, P. Huang, J. He, and R. Fisher, "Supporting ground-truth annotation of image datasets using clustering," in *Pattern Recognition (ICPR), 2012 21st International Conference on*, pp. 1542–1545, 2012.

[42] T. Hastie, R. Tibshirani, and J. J. H. Friedman, *The elements of statistical learning*, vol. 1. Springer New York, 2001.

[43] S. Chib, "Marginal likelihood from the gibbs output," *Journal of the American Statistical Association*, vol. 90, no. 432, pp. 1313–1321, 1995.

[44] P. X. Huang, B. J. Boom, and R. B. Fisher, "Underwater live fish recognition using a balance-guaranteed optimized tree," in *Computer Vision–ACCV 2012*, pp. 422–433, 2013.