Improved Image Annotation and Labelling through Multi-label Boosting

Matthew Johnson and Roberto Cipolla
Department of Engineering,University of Cambridge
mj293@cam.ac.uk,cipolla@eng.cam.ac.uk

Abstract

The majority of machine learning systems for object recognition is limited by their requirement of single labelled images for training, which are difficult to create or obtain in quantity. It is therefore impractical to use methods or techniques which require such data to build object recognizers for more than a relatively small subset of object classes. Instead, far more abundant multi-label data provides a ready means to create object recognition systems which are able to deal with large numbers of classes. In this paper we present a new object recognition system named **MLBoost** which learns from multi-label data through boosting and improves on state-of-the-art multi-label annotation and labelling systems. The system is trained on images with accompanying text and at no time is told which parts of each image correspond to which words, and as such the process is unsupervised. Having once been trained it is able to give segment labels and a list of descriptive words (an annotation) for any novel image.

1 Introduction

The majority of machine learning systems for object recognition are limited by their requirement of single labelled images for training, which are difficult to create or obtain in quantity. This restriction is a major barrier towards building large scale multi-class object recognition systems using these techniques as every class to be learned by the system requires its own set of hand-labelled and sometimes hand-segmented training images. In order to move forward in the field of multi-class object recognition, new techniques must begin to utilize more abundant and easily acquirable types of data. One such type is multi-label data, in the form of images with accompanying text. Among corpora like the Corel image database, newspaper photograph archives with captions, stock advertising photographs and a bevy of other sources there is more than enough data to build the next generation of multi-class object recognizers.

In this paper we present a system called **MLBoost** which is able to learn enough from 1500 or so annotated images of the form shown in figure 1 to perform labelling and annotation on novel images with better results than a state-of-the-art recognizer of the same type [1]. It achieves this by learning the correlation between image segments and the accompanying text in a set of training images. Having learnt this, when given any new image it is able to translate it into words, giving both a labelling for the segments and an annotation for the image as a whole as shown in figure 2.

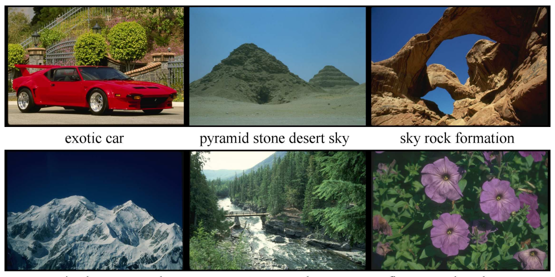


Figure 1: *Typical examples of the training data for* ***MLBoost***. All images were taken from the Corel database and had anywhere from 1 to 5 accompanying keywords which described the image contents. The images were automatically segmented using Normalized Cuts [9] and a feature vector including color, texture and other cues was extracted from each segment for use in the learning process. There was no correspondence given between segments and text.

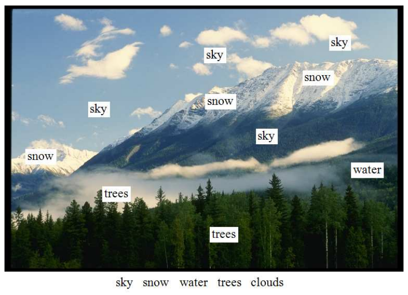


Figure 2: *Output from the* ***MLBoost*** *algorithm*. The word shown is the most probable label for that area of the image, however a full distribution over the vocabulary is produced for each segment. The words below the image are the annotation produced by the algorithm based on the segment labels.

2 Matching Words and Pictures

Visualize a system which sees various images that contain a large segment of solid blue and that always have the word "sky" as one of their labels. Over time, it would be able to learn that the features of those segments and the word "sky" are both different expressions of the same underlying concept, and is then able to translate between them. The algorithms and techniques which pertain to this kind of learning come largely from the machine translation community, and have been adapted for use in computer vision by Barnard et. al [1, 4] who show it to be quite effective at a variety of vision tasks. The initial model they used was inspired by [7], though a full range of models is evaluated in [1] with recent work focusing on latent Dirichlet allocations and probabilistic latent semantic analysis [10, 2].

Barnard et al.'s method in [1] can best be understood by one of the more straightforward models presented there, **I-2**, and it is that model that we will be describing here for the use of comparison and later for improvement. Their method is designed to work with feature vectors, or "blobs", representing the segments of an image. These are related through annotations to associated words which describe the image as a whole. The accuracy of the segmentation method isn't vital provided that it is consistent, allowing the use of automatic segmentation techniques such as normalized cuts [9] to prepare the data, regardless of their lack of accuracy. Once an image has been segmented, a feature vector is extracted for each segment containing color (mean and variance of RGB, Lab, and normalized r/(R+G+B) and g/(R+G+B)), texture (mean and variance of 16 Gabor-like filter responses) and other cues (shape, position, size, etc.) as described in [6].

To statistically link blobs with words, it is assumed that there are hidden factors (concepts) which are responsible for generating *both* the words and blobs associated with that factor. By generating both words and blobs, the concepts can then be used to link the two, learning how they relate. The observations (image and associated text) are assumed to be generated from multiple draws from the hidden factors, as otherwise all possible combinations of words and blobs would need to be modelled. The joint probability of a particular blob, b, and a word w, is modelled as

$$P(w,b) = \sum_c P(w|c)P(b|c)P(c) \tag{1}$$

where c indexes over the concepts, $P(c)$ is the concept prior, $P(w|c)$ is a frequency table, and $P(b|c)$ is a normal distribution over features. The normal distribution over features is assumed to have diagonal covariance to simplify calculation and avoid overfitting. The probability of the observed data, $W \cup B$, given the model, is then:

$$P(W \cup B) = \prod_{b \in B} \left(\sum_c P(b|c)P(c) \right) \prod_{w \in W} \left(\sum_l P(w|c)P(c|B) \right) \tag{2}$$

where W is the set of all annotated words, B is the set of blobs and $P(c|B) \propto \sum_b P(c|b)$, normally limited to the N largest blobs (typically 8 or 10).

This model is fit using the Expectation/Maximization technique [3] treating the concepts as hidden data. This results in the following learning equations for Expectation:

$$P(c|b) \propto P(b|c)P(c) \tag{3}$$

$$P(c|B) \propto \sum_{b \in B} P(b|c)P(c) \tag{4}$$

$$P(c|w,B) \propto P(w|c)P(c|B) \tag{5}$$

and Maximization:

$$P(c) \propto \sum_d \left[\sum_{b \in B} P(c,b) + \sum_{w \in W} P(c|w,d) \right] \tag{6}$$

$$\mu_c = \frac{\sum_b P(c|b)\mathbf{b}}{\sum_b P(c|b)} \tag{7}$$

$$\sigma_c = \frac{\sum_b P(c|b)(\mathbf{b} - \mu_c)(\mathbf{b} - \mu_c)^T}{\sum_b P(c|b)} \tag{8}$$

$$P(b|c) \propto \mathscr{N}(\mu_c, \sigma_c, b) \tag{9}$$

$$P(w|c) \propto \sum_d P(c|w,B) \tag{10}$$

where d indexes the training documents and $\mathscr{N}$ is a multivariate normal distribution.

This is a very effective system but it is limited by the fact that it is forced to perform well overall on the entire data set by the training process. As a result, the concepts which are learned tend to relate textually to the words which occur most often in the data (e.g. sky, water, sun) and visually to the easiest to recognize segments (e.g. solid blue, textured green, yellow circles.) This effect can be seen in figure 4, where the noun-based model predicts the entire image as "sky". While this produces good results for a large subset of the data and thus is optimal behavior for the system, no concepts remain to represent the many words and image segments that are less prevalent. A way to somehow account for this forgotten data would significantly improve performance of the algorithm.

3 A Multi-label Boosting Framework

Boosting combines several weak learners to create a single system which is better than the sum of its parts. The iterative process tells each subsequent learner to concentrate on the subset of data which has yet to be learned by the previous learners. In this way, it is an ideal solution to the weakness of the system just presented; however, no boosting system has ever been built which is able to boost multi-label learners of this variety. In this paper we present such a system, **MLBoost**.

3.1 Multi-class Boosting

In its most basic form, Adaboost deals with binary classification (i.e. something is/isn't a member of a class). However, Freund and Schapire provide two multi-class versions of Adaboost in [5]. Our multi-label framework is an extension of the second version, which dealt with learners that were evaluated in terms of pseudoloss. The concept of pseudoloss is quite well suited to dealing with the data ambiguity problem at hand. It was originally defined as

$$\text{ploss}_q \doteq \frac{1}{2}\left(1 - h(x_i, y_i) + \sum_{y \neq y_i} q(i,y)h(x_i,y)\right) \tag{11}$$

where q is a label weighting function, x_i is a data input, y_i is the correct label for that data, and h is a hypothesis returned by a weak learner which assigns a certainty value between 0 and 1 for each label given the input. The label weight function, q, deserves some mention. This function measures the degree to which the weak learner mistakes x_i for being a member of another class. It is calculated such that $\sum_{y \neq y_i} q(i,y) = 1$. In this way, if the certainty sum for every label (except the correct label) is 1 then the pseudoloss is 1, and if the classifier is perfect (the correct label has a certainty of 1 with all others 0) the pseudoloss is 0. The place of q in the algorithm is to guide the learner towards learning certain classes more thoroughly than others, which enables the boosting system to concentrate on classes which are hard to classify and underrepresented.

Our situation is slightly different from that of this initial formulation. Whereas the original deals with data items with single labels, we are dealing with data items which have multiple labels. The learner must maximally predict the correct labels and minimally predict the incorrect labels, and we have modified the pseudoloss in a straightforward manner to reflect this:

$$\text{ploss}_q \doteq \frac{1}{2}\left(1 - \frac{\sum_{y \in Y_i} h(x_i,y)}{|Y_i|} + \sum_{y \in Y - Y_i} q(i,y)h(x_i,y)\right). \tag{12}$$

3.2 Multi-Label Weak Learners

Using this modified pseudoloss as a basis, our modified form of Adaboost, named **MLBoost**, is presented as algorithm 1. It should be noted that there are two novel and key differences between this algorithm and Adaboost.M2. As previously mentioned, the first difference is the modification to the calculation of pseudoloss. The second difference is that the recalculation of the weight vector has been modified to incorporate multi-label output. The weight vector is updated based on whether a particular label is a member of the symmetric difference between the correct annotation for a particular input, Y_i, and the annotation of that input, A_i, which is the set of elements y for which h_t is maximally confident, as done by Schapire and Singer in [8]. For all labels y which are incorrectly given by the hypothesis in the annotation A_i, the document specific weight, $w_{i,y}$ increases by a factor of K, the training speed constant. For all other y, the weight decreases by a factor of K. In this way, documents which are consistently labelled correctly decrease in training significance exponentially while documents which are consistently annotated incorrectly

Algorithm 1 MLBoost

```
1: Input:
   1. documents $\langle (x_1,Y_1),\ldots,(x_N,Y_N)\rangle$ with labels $Y_i \subset Y = 1,\ldots,k$
   2. distribution $D$ over the documents
   3. multi-label weak learning algorithm MultiWeakLearn
   4. integer $T$ specifying number of iterations
   5. training speed constant K
2: Initialize the weight vector: $w^1_{i,y} = 1$
3: for $t = 1$ to $T$ do
4:     Set $W^T_i = \sum_{y \notin Y} w^t_{i,y}$;
           $$q_t(i,y) = \frac{w^t_{i,y}}{W^t_i}$$
       for $y \notin Y_i$; and set
           $$D_t(i) = \frac{W^t_i}{\sum_{i=1}^N W^t_i}$$
5:     Call MultiWeakLearn, providing it with the distribution $D_t$ and label weighting
       function $q_t$; get back hypothesis $h_t : X \times Y \to [0,1]$.
6:     Set $\varepsilon_t$ equal to the pseudoloss of $h_t$ (see equation 12)
7:     Set $\beta_t = \frac{\varepsilon_t}{(1-\varepsilon_t)}$.
8:     Produce an annotation set $A_i$ for each $x_i$ where $|A_i| = |Y_i|$ and contains the most
       likely labelling elements $y$ with regard to $h_t$.
9:     Set the new weights vector to be
           $$w^{t+1}_{i,y} = \begin{cases} K w^t_{i,y} & y \in A_i \ominus Y_i \\ \frac{w^t_{i,y}}{K} & \text{otherwise} \end{cases}$$
       for $i = 1,\ldots,N, y \in Y - Y_i$.
10: end for
11: Output the hypothesis
           $$h_f(x,y) = \sum_{t=1}^{T} \left(\log \frac{1}{\beta_t}\right) h_t(x,y)$$
    .
```

increase in training significance. Also, those which are consistently annotated incorrectly with the same words increase in training significance exponentially. In our experiments we used a value for K of $N^{1/T}$, where N is the number of training documents. Our choice of K was made such that it was difficult for a single document to dominate all training by being annotated incorrectly every time with the same word until the end of the training run.

In keeping with the original form of this algorithm, the learner to be boosted, **MultiWeakLearn**, need only produce hypotheses with pseudoloss consistently below or above $1/2$ with respect to the distribution over the documents D and the label weighting function q with which it was presented. However, **MultiWeakLearn** must be able to learn from multi-label data. Because of this, the candidate learning systems tend to be far more complex than in the supervised case, where often simple decision stumps are all that is required [11]. Previous attempts at boosting more complex learners for speech recognition [12] have shown that boosting is still useful in these circumstances, though the resulting array of hypotheses can be time- and space-intensive for calculation and storage, respectively. In our initial exploration of this topic we disregarded this aspect in favor of exploring the usefulness of the technique, however future work will need to concentrate on ways to improve performance and decrease time and space complexity.

4 A Boostable Multi-Label Vision System

Before we could use this boosting framework to create a vision system, we had to produce a candidate for **MultiWeakLearn**. Our candidate for **MultiWeakLearn** in this paper is a novel modification to the method used by Barnard et. al in [1] and described in section 2. In the learning equations, we make several key insertions to bias the learning towards subsets of the data and against particular labels as determined by the distribution D and label weighting function q provided by the boosting system as described in algorithm 1.

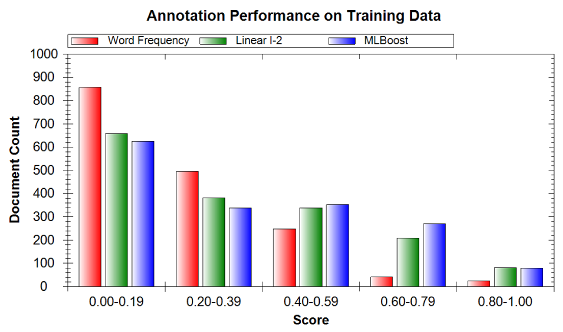
We use D to influence the clustering of blobs by giving more weight to blobs in emphasized documents during Maximization. $D(i)$, is added as an addition weight to $P(c|b)$ in equations 7 and 8. We use q, or rather the value $1 - q_t(w,i)$ to deemphasize incorrect labels when calculating $P(c|w,B)$ in the Expectation (specifically, equation 5).

This system provides a hypothesis to the booster which calculates a vocabulary distribution for a document using a novel weighted voting scheme. We introduced this scheme due to problems inherent in the original annotation system provided by Barnard et. al, which would produce a $P(w|B)$ as

$$P(w|B) \propto \sum_{b} P(w|c)P(c|B). \tag{13}$$

Imagine an image of a plane flying in the sky. The blue segments of sky give a large contribution to the solid blue, “sky” concept. The one or two segments depicting the plane give a smaller contribution to the textured gray, “plane” concept. The resulting annotation is heavily biased towards the first concept and contains words like “sky” and “water” but not “plane”, even though that segment was correctly labelled.

We propose the following solution: for each blob, the words in the vocabulary are ranked with regard to their likelihood as a labelling for that blob as determined by $P(w,b)$. Then a vote is cast for each word with weight $v_w = \frac{1}{2^{r_w}}$, where r_w is the rank of a particular word (i.e. $r_w = 0$ for the first ranked y and $|Y| - 1$ for the last). The resulting collection of votes is then normalized to produce a distribution $P(w|B)$ over the vocabulary.



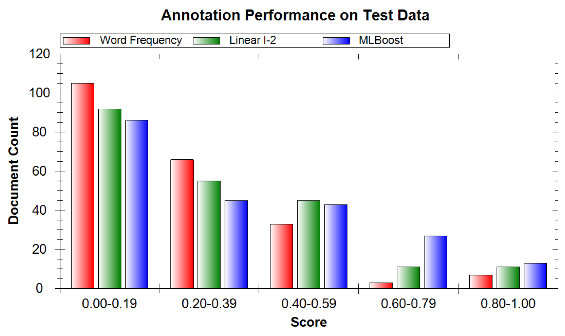


Figure 3: *Annotation Performance on training and test data.* The training subset consisted of 1667 images, or 90% of the data, with the test subset consisting of the remaining 10%, or 214 images. The data for word frequency comes from simply producing an annotation of the five most common words in the training data set for each image. The data from **I-2** comes from our implementation of the linear form of the model in [1]. As expected, **MLBoost** has fewer "bad" annotations (i.e. the first two bins), and the stressing of overall performance in the algorithm introduced by taking the mean of the confidence scores for the correct labels results in more "good" annotations (i.e. the last three bins), particularly those which are almost entirely correct. Score is determined as the number of correctly annotated words from an algorithm divided by the number of keywords for an image.

5 Results

We trained the algorithm using a subset of 1881 images from the Corel database. The state-of-the-art system used for comparison is an unmodified implementation of model **I-2** from [1] with linear topology. Our system, **MLBoost**, was implemented as shown in algorithm 1. The models were evaluated on two tasks: annotation and labelling. Each was trained on roughly 90% of the data, or 1667 images, and a test set of the remaining 214 images was used for evaluation of the model's generalization ability.

5.1 Annotation

Annotation provides a straightforward means to determine the effectiveness of the models under study. The baseline for performance is difficult to set; however, in [1] a word frequency distribution extracted from the training data has proven to be a rough equivalent of the "oblivious" algorithms used elsewhere in the literature for a similar purpose. This annotation simply consists of the top N words according to usage, where N is the size of the correct annotation for an image. Due to the nature of the Corel database, the keywords provided as correct annotations in our data contain many of the same words (e.g. "rock", "sky", "close-up"), allowing such a method of annotation to do quite well. To perform better than this, a learner must be gaining knowledge from co-occurrence with image data, not simply word usage statistics, and as such it serves well as a baseline.

Annotations are produced as described in section 4. The annotation performance is shown in figure 3 for annotations produced using the word frequency and the two models. As expected, **MLBoost** has fewer "bad" annotations (i.e. the first two bins), and the stressing of overall performance in the algorithm introduced by taking the mean of the confidence scores for the correct labels results in more "good" annotations (i.e. the last three bins, particularly the second to last bin.)

5.2 Labelling

Labelling is a difficult task as it is essentially object recognition. The segments produced by normalized cuts tend not to be optimal for the task, though this problem is not endemic to the algorithm as automatic segmentation remains an open problem. That aside, the system produced by Barnard et al. is capable of performing rather well on the task, and as such we can boost that performance. Automatic evaluation of the task is impossible, as the matching between segments and keywords is an unknown quantity. In order to do a thorough evaluation of labelling performance, hand labelling with multiple labels of each segment in every image would have to be performed. While such an evaluation is desirable, it is not within the scope of this paper.

Presented in figure 4 are 3 examples each from **I-2** on the left and from **MLBoost** on the right. The automatic segmentations of the images are shown in red, and the top-scoring label for each of the 10 largest segments is displayed. The five words at the bottom of each image are the annotating words produced for the image ordered from left to right in terms of likelihood, with bold font indicating those which are correct. It should be noted that the correct annotations did not always have 5 words, though 5 was the maximum. All five maximally-predicted words are shown to give insight into the nature of the distribution. Note that **I-2** is achieving annotation performance by taking the easiest path in terms of segment labelling, whereas MLBoost is correctly labelling the segments and achieving equal or better annotation performance on the same image input.

6 Conclusion

The **MLBoost** algorithm has proven to be effective at improving the performance of a candidate weak multi-label learner and has shown better results at both image annotation and labelling. As it is able to learn from multi-label data, it is also able to perform multi-class object recognition in addition to input labelling, and as such is a promising framework on which to build next-generation object recognition systems.

It is unclear without further exploration whether the improvement boosting provides over the **I-2** model is due to that model's inherent weaknesses which may not be present in better statistical modelling paradigms such as the others presented in [1] or probabilistic semantic analysis as seen in [10]. Future research will concentrate on whether boosting indeed provides a boon with these more complex systems.

We are also interested in exploring the uses of boosting as a method of combining different kinds of learners. For example, a color-based learner and a texture-based learner could each concentrate on the classes which are easiest to identify in their space and the booster could choose one or the other at each stage of the algorithm according to performance. It could also choose between different modelling paradigms based on which best models a particular region of the target space.

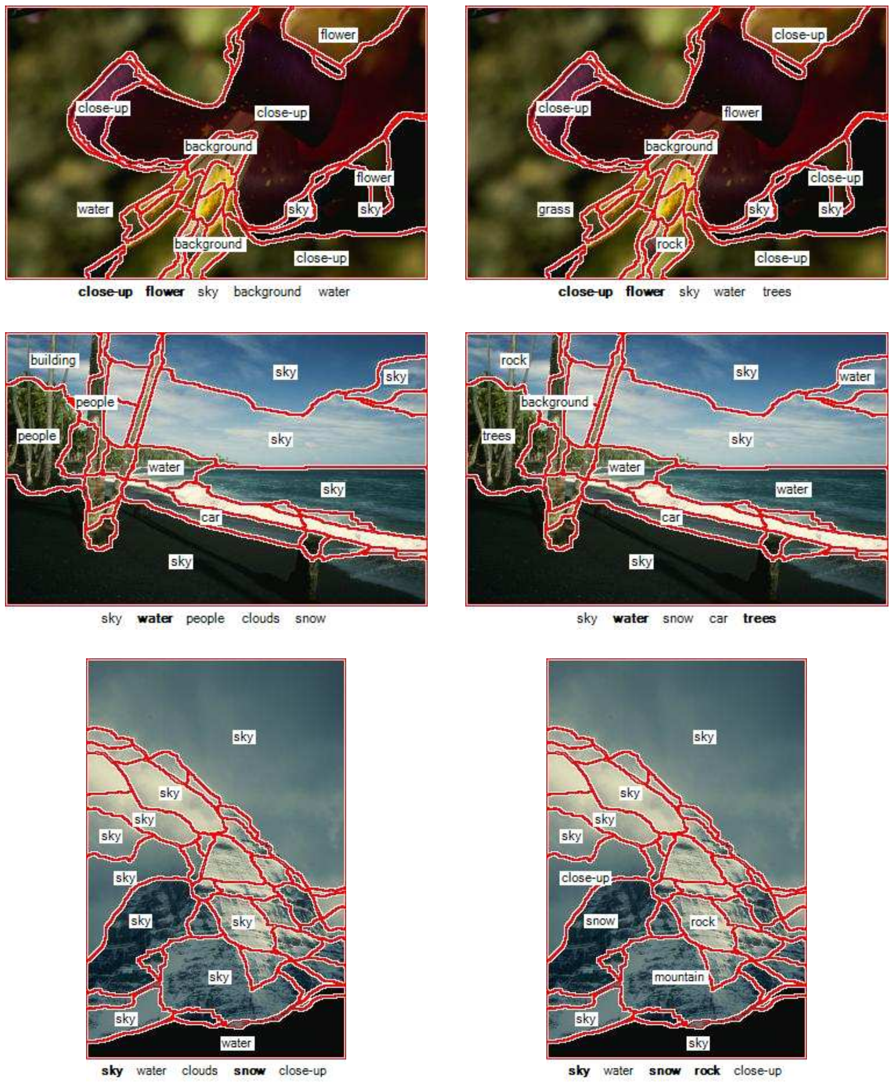


Figure 4: *Labelling performance.* The labellings on the left come from model **I-2**, on the right from **MLBoost**. The automatic segmentations of the images are shown in red, and the top-scoring label for each of the 10 largest segments is displayed. The five words at the bottom of each image are the annotating words produced for the image ordered from left to right in terms of likelihood, with bold font indicating those which are correct. Note that **I-2** is achieving annotation performance by taking the easiest path in terms of segment labelling, whereas MLBoost is correctly labelling the segments and achieving equal or better annotation performance on the same image input.

References

[1] Kobus Barnard, Pinar Duygulu, Nando de Freitas, David Forsyth, David Blei, and Michael I. Jordan. Matching words and pictures. *Journal of Machine Learning Research*, 3:1107–1135, 2003.

[2] David M. Blei. *Probabilistic models of text and images*. PhD thesis, University of Berkeley, 2004.

[3] A.P. Dempster, N.M. Laird, and D.B. Rubin. Maximum likelihood from incomplete data via the em algorithm. *Journal of the Royal Statistical Society. Series B (Methodological)*, 39(1):1–38, 1977.

[4] Pinar Duygulu, Kobus Barnard, Nando de Freitas, and David Forsyth. Object recognition as machine translation: Learning a lexicon for a fixed image vocabulary. In *Proceedings of ECCV2002*, Copenhagen, Denmark, May 2002.

[5] Yoav Freund and Robert E. Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of Computer and System Sciences*, 55(1):119–139, 1997.

[6] Prasad Gabbur. Quantitative evaluation of feature sets, segmentation algorithms and color constancy algorithms using word prediction. Master's thesis, University of Arizona, 2003.

[7] Thomas Hofmann. The cluster-abstraction model: Unsupervised learning of topic hierarchies from text data. In *Proceedings of the International Joint Conference in Artificial Intelligence*, 1999.

[8] Robert Schapire and Yoram Singer. Improved boosting algorithms using confidence-rated predictions. *Machine Learning*, 37(3):297–336, 1999.

[9] J. Shi and J. Malik. Normalized cuts and image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22:888–905, 2000.

[10] Joset Sivic, Bryan C. Russell, Alexei A. Efros, Andrew Zisserman, and William T. Freeman. Discovering object categories in image collections. Technical Report AIM-2005-005, Massachusetts Institute of Technology, February 2005.

[11] Antonio Torralba, Kevin P. Murphy, and William T. Freeman. Sharing features: efficient boosting procedures for multiclass object detection. In *Proceedings of CVPR '04*, 2004.

[12] Geoffrey Zweig and Mukund Padmanabhan. Boosting gaussian mixtures in an lvcsr system. In *Proceedings of ICASSP 2000*, 2000.