

RATE-CONSTRAINED CLASSIFICATION USING BFOS ALGORITHM

Vijay Chandrasekhar, Shang-hsuan Tsai and Zhi Li

EE378 Class Project, Spring 2007/2008

ABSTRACT

We study the problem of rate-constrained classification based on local features. Image classes are represented as Gaussian mixture of features. Features from a query image are quantized with a tree-structured vector quantizer (TSVQ) and transmitted over a channel for the purpose of classification. We define a cluster entropy functional to indicate how discriminative each quantization cluster is. With the end goal of image classification, we optimally prune the TSVQ tree using the BFOS algorithm, and study the trade off between the rate and classification error. At a given rate, our scheme provides a lower classification error, than a scheme that prunes the TSVQ tree with average distortion as the metric. We then study various classification criteria and propose a maximum a posteriori (MAP) rule that outperforms other heuristic rules. We further justify our proposed methods with the analysis of error probabilities.

1. INTRODUCTION

Our work falls under the category of content-based image retrieval (CBIR) [1, 2]. We want to recognize real-world images from a set of database images, but we want to do it with a rate constraint.

1.1. Related Work

Typically, histogram of gradient based features are extracted from a query image. The extracted features are then compared to a database of features. Research on robust local descriptors is very active. Recent examples include, but are not limited to, SIFT by Lowe [3], GLOH by Mikolajczyk and Schmid [4], and SURF by Bay et al. [5].

Recent work in object-based image retrieval uses tree-structured vector quantizers (TSVQs) to search for similar images in very large image collections [6, 7, 8]. TSVQs are commonly known as Scalable Vocabulary Trees (SVTs) in the computer vision community. The nodes of the TSVQ are referred as ‘visual words’. The problem of image retrieval is then analogous to one of text retrieval, with the words in the former case being local SIFT or SURF features extracted from images, and quantized with a TSVQ.

The vocabulary tree is built using hierarchical K-means clustering. A large set of representative descriptor vectors are

used in the unsupervised training of the tree. K defines the branch factor (i.e. number of children of each node) of the tree. First, an initial K-means process is run on the training data, defining K cluster centers. The training data is then partitioned into K groups, where each group consists of the descriptor vectors closest to a particular cluster center. The same process is then recursively applied. The path down the tree to any node can be encoded by a single integer, which is then used to represent that node or quantization cluster. Hierarchical K-means, compared to traditional K-means, is a scalable approach as it allows us to add features to the tree without having to rebuild the tree with the addition of new features.

Each feature from the database is quantized to a leaf node of the tree. Each node of the tree stores a pointer to all the features that were quantized to the cluster. The quantized codewords, thus, point back to the images from which they were extracted. This indexing is used to determine which image class an incoming set of set of query features belongs to. Each query feature is quantized greedily to a leaf node in the TSVQ. The image with the closest distribution of features to the query image is considered to be the most likely match.

The computer vision community uses TSVQs, primarily for the purpose of classification. We are interested in using TSVQs for compressing features as well. We assume that the TSVQ used for classification at the server (receiver) is also available at the transmitter.

Extensive work has been done on how to prune TSVQs optimally to meet a rate-distortion criterion. The BFOS algorithm is commonly used for pruning TSVQs [9].

1.2. Paper outline

Section 2 discusses the formulation of the the rate-constrained classification problem. Section 3 provides a brief overview of the BFOS algorithm. Section 4 provides the details of our proposed algorithm. Tree building, pruning and the different criteria used for classification are discussed in this section. Section 5 discusses the experiment set-up and the results.

2. PROBLEM FORMULATION

We address a rate-constrained classification problem based on local features. Consider the application of building recogni-

tion through mobile phone snapshots. First of all, a query image of a building (e.g. Packard) is taken. This is followed by extraction of multiple high-dimensional local features from the image (on the mobile phone). Note that we do local feature extraction instead of the traditional eigen-image approach. This is in accordance with the recent popular use of SIFT/SURF features in the computer vision community. For each local feature, we want to quantize it and encode into a few bits as possible and send to the server through a bandwidth-limited wireless link. At the server side, the decoded query features are matched against a database of buildings (e.g. Gates, Packard, Hewlett).

One distinguished feature of our formulation is that we explicitly take the ‘rate’ as a factor in the classification problem. In general, there is a trade-off between the retrieval accuracy (in terms of classification error) and the rate of the query. We can expect that the fewer bits of representation, the higher the classification error rate.

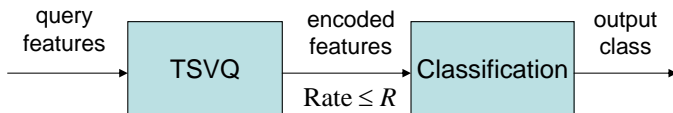


Fig. 1. The proposed solution framework for the rate-constrained classification problem.

Our solution framework can be described by Fig. 1. We use tree-structured vector quantization (TSVQ) since it allows easy scaling to very large databases. For each extracted features of the query image, we quantize it to a leaf node of the tree. All the quantized features are then jointly entropy-coded subject to a rate constraint R . (The tree is trained offline using a bunch of training data with labels and made known to the mobile phone). The encoded features are sent over to the server, where a classification decision is made on which class these feature vectors are generated from.

We consider incorporating the BFOS algorithm into our framework for several reasons. First, it naturally integrates with the TSVQ paradigm. Second, the BFOS pruned trees allow the quantizer to operate at different rates, thus facilitating rate control.

Also note that the original BFOS algorithm was first developed for classification and regression problems in [10], and later extended to source coding in [9]. Our problem formulation differs from the original BFOS in the following ways. First, since we deal with multiple local features, the original BFOS pruning criterion based on classification errors cannot be applied to our problem. New pruning criteria need to be found. Second, the original BFOS problem is constrained by the number of nodes in the tree whereas our problem is rate-constrained.

3. BFOS ALGORITHM

Here, we provide a brief overview of the BFOS algorithm. Refer to [9] for the exact details of the algorithm. The BFOS algorithm was initially introduced in the context of classification and regression trees [10], but extended to variable rate TSVQ by [9]. For TSVQs, the algorithm begins with an initial tree, and prunes it back until it reaches the subtree with the fewest number of leaves that achieves a given rate distortion trade-off. For variable-rate TSVQs, the algorithm serves to trace out the convex hull of the operable distortion-rate function.

The BFOS algorithm can be applied with several design criteria. The design criteria are called tree functionals. If the tree functionals are monotonic increasing or decreasing, the BFOS algorithm optimally prunes the tree. Number of leaves, number of nodes, entropy, average distortion, conditional entropy are all examples of tree functionals.

A system designer first has to decide which functionals to trade-off. We are interested in the trade off between rate and classification error, and propose tree functionals, to that effect.

4. RATE-CONSTRAINED CLASSIFICATION USING BFOS

4.1. Tree Building

The merit of TSVQ lies in its ability to scale as the amount of data increases. The tree can be grown into deeper levels to accommodate the growing number of data points. In the computer vision community, [7, 8] have shown how TSVQ trees can be used to represent millions of features.

In this work, we explore two different approaches to building the initial quantization tree. The first tree is a hierarchical K-means tree. Given a branching factor and maximum depth of the tree, we apply the K-means algorithm recursively to generate the hierarchical data structure. The second tree we examine is a K-D tree. The K-D tree is a tree which bisects data over all dimensions iteratively. We would like to apply the BFOS pruning algorithm to both tree structures.

4.2. Tree Pruning

After the initial tree is chosen, we want to optimally prune the tree successively to produce various subtrees with different rates and classification error probabilities.

Let us first define some notations. Consider we have a (possibly pruned intermediate) tree with $|T|$ leaf nodes, each corresponding to a quantization cluster. For an incoming feature, we quantize it to one of the $|T|$ clusters. Let T be a random variable representing the cluster. Each feature belongs to one of the $|C|$ classes. Let C be a random variable representing the class. We can define the following probabilities.

- $Pr(T = t) = p(t)$ for $t = 1, \dots, |T|$, the probability that a feature is quantized to cluster t .
- $Pr(C = c) = p(c)$ for $c = 1, \dots, |C|$, the prior probability of class c .
- $Pr(C = c|T = t) = p(c|t)$ for $c = 1, \dots, |C|$ and $t = 1, \dots, |T|$, the probability that a feature quantized to cluster t belongs to class c .
- $Pr(T = t|C = c) = p(t|c)$ for $c = 1, \dots, |C|$ and $t = 1, \dots, |T|$, the probability that a feature in class c is quantized to cluster t .

Given a bunch of training features and a fixed tree structure, we can easily compute the above quantities.

In the source coding formulation of BFOS in [9], the design goal is to minimize the quantization distortion with the rate constraint. Thus the two monotonic affine tree functionals are chosen to be $u_1(t) = l(t)p(t)$, the average codeword length of t , and $u_2(t) = \delta(t)p(t)$, the average distortion of t . In our problem, our objective is to minimize the retrieval error with the rate constraint. Therefore, we have the same functional u_1 as in [9]. However, the second functional warrants further discussion.

In [9], the BFOS algorithm guarantees optimal pruning results in terms of a rate-distortion tradeoff. If the quantized features minimize distortion with respect to the original distortion, we expect good classification results. Note however that ‘distortion’ is not the ultimate goal of our problem. The ultimate goal is to minimize classification error. Can we do better by finding a functional u_2 that more directly relates to classification error? Here, we present a simple counter example where minimizing distortion does not automatically guarantee minimizing classification error. Refer to Fig. 2. If we have two classes of data following distributions as shown here, the criterion of minimizing classification error leads to the dashed horizontal line being the decision boundary. However, if distortion criterion is used, the decision boundary would be the solid vertical line. This example is somewhat extreme, but it tells us in general, that if our goal is to minimize the classification error, given the class labels, we can do better than merely minimizing an average distortion.

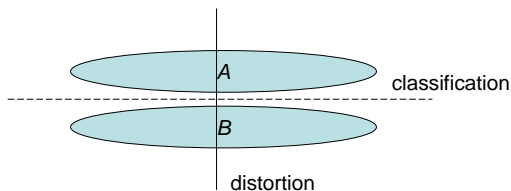


Fig. 2. A counter-example where minimizing distortion does not automatically guarantee a minimized classification error.

$p(c t)$	$o(t)$
[0.9 0.1]	0.2
[0.5 0.5]	1
[0.5 0.4 0.1]	1.2
[0.33 0.33 0.33]	2

Table 1. An example of computing cluster overlap.

How do we choose u_2 so that it directly relates to classification error? A simple heuristic is that if features of multiple classes are quantized into one cluster, this cluster would be non-representative of any of the classes, and hence not useful for classification. Furthermore, the worst case is when all the classes are equally likely in the cluster (i.e. $p(c|t)$ is equal for all c), because if any of the classes has a higher probability, this cluster can be somewhat more representative for that class. We propose two reasonable metrics to reflect the above observations.

4.2.1. Cluster overlap

We define the *overlap* of cluster t as

$$o(t) = \sum_n \sum_{m \neq n} \min(p(n|t), p(m|t)). \quad (1)$$

Then we can use the average overlap as the functional $u_2(t)$:

$$u_2(t) = o(t)p(t). \quad (2)$$

In the extreme case, if cluster t only has features from one class, then $o(t) = 0$. If cluster t contains features from K classes with evenly distributed $p(c|t)$, then $o(t) = K - 1$. We can see that this quantity well captures the above-mentioned two heuristics. Table 1 shows some example of cluster conditional probability and the overlap computed.

4.2.2. Cluster entropy

Another metric we can use to capture the heuristics is the *conditional entropy* of cluster t , defined as:

$$H(C|t) = - \sum_c p(c|t) \log p(c|t). \quad (3)$$

The cluster entropy measures the ‘uncertainty’ after given the cluster is t . When cluster t only has features from one class, $H(C|t) = 0$. If cluster t contains features from K classes with evenly distributed $p(c|t)$, then $H(C|t) = \log K$. Note that entropy is a concave functional. In the tree pruning algorithm, we would rather have a convex functional (this is related to a practical issue of BFOS algorithm. If a concave functional is used, it would produce fewer operational points in the trade-off curve). One method to overcome this issue is to use $2^{H(C|t)}$ instead of $H(C|t)$. An intuitive interpretation

is that $2^{H(C|t)}$ captures the average number of classes corresponding to cluster t . Then we can define the functional $u_2(t)$ as:

$$u_2(t) = 2^{H(C|t)} p(t). \quad (4)$$

Qualitatively, both metrics *overlap* and *entropy* capture the essence of the two observations. A tree pruning with respect to any one of the two criteria is expected to produce clusters that contain as few number of classes as possible, and distributions as uneven as possible. This would lead to more distinguished class distributions $p(t|c)$, thus improve the classification results. In section 4.4, we will show that the error probabilities of classification is related to the divergence of $p(t|c)$ between different classes. This would further justify our proposed metrics.

Finally, it can be shown that both metrics are linear and monotonically increasing as the tree is pruned. Therefore, they are proper functionals that can be used in the BFOS algorithm [9].

4.3. Classification Criteria

Given some quantized and coded features, what is the optimum criterion to decide which class they belong to? In this section we first look at various classification criteria that are widely used in the computer vision community. We then formulate classification as a signal detection problem and propose a classification rule based on the *maximum a posteriori* (MAP) criterion.

4.3.1. L_1 -norm and variants

With an incoming query of N quantized features t_1, t_2, \dots, t_N , we construct the histogram (i.e. empirical pmf) using the N features, and compare it with pmf of each class using L_1 -norm distance. The query is classified to the class leading to the least distance.

A heuristic improvement is that when constructing the pmf, we should also account for the internal tree nodes along the hierarchical quantization path. We refer to this approach as L_1 -norm with internal nodes.

4.3.2. L_2 -norm

This approach is similar to the one above, except than when computing the distance between two pmfs, we use the L_2 -norm instead of L_1 .

4.3.3. Symmetric KL Divergence

A common distance metric used for two probability distributions is the Symmetric Kullback-Leibler divergence. Which is defined as below for two different pmfs P, Q as:

$$\begin{aligned} D_{symKL}(P, Q) &= D_{KL}(P||Q) + D_{KL}(Q||P) \\ &= \sum_{i=1}^{|T|} P(i) \log \frac{P(i)}{Q(i)} + \sum_{i=1}^{|T|} Q(i) \log \frac{Q(i)}{P(i)}. \end{aligned} \quad (5)$$

4.3.4. χ^2 Distance

Another distance metric is the χ^2 distance, as in [11]. The distance between two histograms with $U = (u_1, u_2, \dots, u_{|T|})$ and $W = (w_1, w_2, \dots, w_{|T|})$ is defined as:

$$D_{\chi^2}(U, W) = \frac{1}{2} \cdot \sum_{i=1}^{|T|} \left(\frac{(u_i - w_i)^2}{u_i + w_i} \right). \quad (6)$$

4.3.5. MAP Rule

We note that this classification problem can also be formulated as a signal detection problem. This is shown in Fig. 3. The message c is communicated through a noisy channel with conditional probability $p(\mathbf{t}|c)$. Based on observations $\mathbf{t} = [t_1, \dots, t_N]$, the detector makes an decision on \hat{c} .

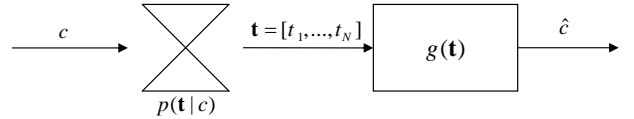


Fig. 3. Retrieval as a signal detection problem.

A useful detection rule is MAP, i.e. we want to find

$$\hat{c} = g(\mathbf{t}) = \arg \max_c p(c|\mathbf{t}). \quad (7)$$

Assume that the observations t_1, \dots, t_N are conditionally i.i.d. given c , i.e.,

$$p(\mathbf{t}|c) = \prod_{i=1}^N p(t_i|c), \quad (8)$$

we can then express

$$p(c|\mathbf{t}) = \frac{p(\mathbf{t}|c)p(c)}{p(\mathbf{t})} = \frac{\prod_{i=1}^N p(t_i|c)p(c)}{p(\mathbf{t})}. \quad (9)$$

The MAP rule is then

$$\hat{c} = \arg \max_c \prod_{i=1}^N p(t_i|c)p(c). \quad (10)$$

4.4. Error Probability Analysis

First we analyze the probability $p(\mathbf{t}|c)$. Assume N is large, and let \tilde{c} be the true class where \mathbf{t} is drawn from, i.e. $\mathbf{t} \in A_{p(t|\tilde{c})}^N$, the typical set with respect to $p(t|\tilde{c})$. We have:

$$\begin{aligned} \log p(\mathbf{t}|c) &= \sum_{i=1}^N \log p(t_i|c) \\ &= N \cdot \sum_{\tilde{c}} p(t|\tilde{c}) \log p(t|c) \\ &= N \cdot \left(- \sum_{\tilde{c}} p(t|\tilde{c}) \log \frac{p(t|\tilde{c})}{p(t|c)} + \sum_{\tilde{c}} p(t|\tilde{c}) \log p(t|\tilde{c}) \right) \\ &= N \cdot (-D(p(t|\tilde{c})||p(t|c)) - H(T|\tilde{c})), \end{aligned} \quad (11)$$

where the second equation follows from the law of large numbers (LLN), and $D(\cdot||\cdot)$ is the Kullback-Leibler divergence. Then we have:

$$p(c|\mathbf{t}) = \frac{p(c)}{p(\mathbf{t})} 2^{-N \cdot D(p(t|\tilde{c})||p(t|c)) - N \cdot H(T|\tilde{c})}. \quad (12)$$

In particular, for the true class $c = \tilde{c}$,

$$p(\tilde{c}|\mathbf{t}) = \frac{p(\tilde{c})}{p(\mathbf{t})} 2^{-N \cdot H(T|\tilde{c})}. \quad (13)$$

Now we can write the average probability of error as

$$\begin{aligned} P_e &= \sum_c p(c) Pr(\hat{C} \neq c | C = c) \\ &= \sum_c p(c) \sum_{\mathbf{t}} p(\mathbf{t}|c) Pr(\hat{C} \neq c | \mathbf{t}, C = c) \\ &= \sum_c p(c) \sum_{\mathbf{t}} p(\mathbf{t}|c) Pr(\hat{C} \neq c | \mathbf{t}) \\ &= \sum_c p(c) \sum_{\mathbf{t} \in A_{p(t|c)}^N} p(\mathbf{t}|c) Pr(\hat{C} \neq c | \mathbf{t}), \end{aligned} \quad (14)$$

where the third equation follows from the conditional independence between C and \hat{C} given \mathbf{t} (since \hat{C} is a function of \mathbf{t}). The fourth equation follows from the fact that $p(\mathbf{t}|c)$ is small for \mathbf{t} outside the typical set $A_{p(t|c)}^N$. For $\mathbf{t} \in A_{p(t|c)}^N$, $Pr(\hat{C} \neq c | \mathbf{t})$ is:

$$\begin{aligned} &Pr(\hat{C} \neq c | \mathbf{t} \in A_{p(t|c)}^N) \\ &= I(p(d|\mathbf{t}) > p(c|\mathbf{t}) \text{ for any } d \neq c | \mathbf{t} \in A_{p(t|c)}^N) \\ &= 1 - I(p(d|\mathbf{t}) \leq p(c|\mathbf{t}) \forall d \neq c | \mathbf{t} \in A_{p(t|c)}^N) \\ &= 1 - I\left(D(p(t|c)||p(t|d)) \geq -\frac{1}{N} \log \frac{p(c)}{p(d)} \forall d \neq c\right), \end{aligned} \quad (15)$$

where $I(\cdot)$ is the indicator function.

We can then express P_e as:

$$\begin{aligned} P_e &= \sum_c p(c) \sum_{\mathbf{t} \in A_{p(t|c)}^N} p(\mathbf{t}|c) Pr(\hat{C} \neq c | \mathbf{t}) \\ &= \sum_c p(c) \sum_{\mathbf{t} \in A_{p(t|c)}^N} 2^{-N \cdot H(T|c)} \\ &\quad \cdot \left(1 - I\left(D(p(t|c)||p(t|d)) \geq -\frac{1}{N} \log \frac{p(c)}{p(d)} \forall d \neq c\right)\right) \\ &= 1 - \sum_c p(c) I\left(D(p(t|c)||p(t|d)) \geq -\frac{1}{N} \log \frac{p(c)}{p(d)} \forall d \neq c\right). \end{aligned} \quad (16)$$

In the limiting case, assume the prior probability $p(c)$ is uniform, and the conditional distribution $p(t|c)$ is different for all classes, then the probability of error tends to 0. This agrees with our intuition that if we have more features available, we have more information, thus lower classification errors. This expression also justifies our overlap and entropy criterion in Section 4.2. To minimize the error probability, we need to maximize the probability divergence $D(p(t|c)||p(t|d)) \forall d \neq c$. This suggests that within each cluster c , we need to keep the probability ‘overlap’ low, or keep the ‘uncertainty’ (measured by conditional entropy) low, which agrees with our heuristic.

5. EXPERIMENTAL RESULTS

We evaluate the tree building, tree pruning and classification algorithms proposed in Section 4.

5.1. Experiment Setup

We perform our experiments on synthetic data. Each image class is represented as a Gaussian mixture model (GMM) with 7 Gaussians. The means and covariance matrices for each GMM are chosen at random. The feature data is two dimensional for better visualization purpose. A typical mixture of features is shown in Fig. 4, where different colors represent different classes. An error is reported if the top match based on the classification criterion is incorrect. The results presented are averaged over 100 runs of randomly generated training and testing data.

5.2. Results and Discussions

In Fig. 5, we show the classification results of the hierarchical K-means tree and the K-D tree using cluster entropy as the pruning criterion and MAP as the classification criterion. We observe that the two quantization trees have similar performance at high bit-rates. At low bit-rates, the K-mean tree performs a little better than the K-D tree. This is due to the fact that the K-means algorithm adapts to data statistics, while the K-D tree uses fixed lattice quantization. Overall, we conclude that the K-means tree performs slightly better, and use it for the rest of our experiments.

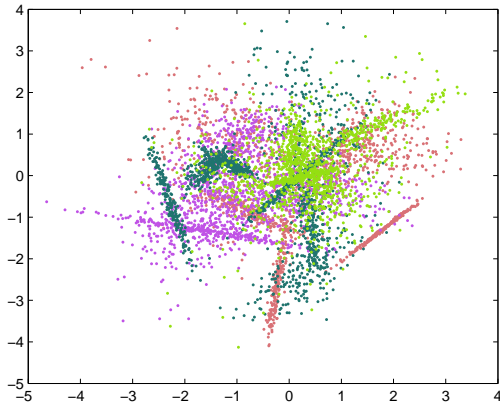


Fig. 4. Typical Gaussian mixture generated for experiments.

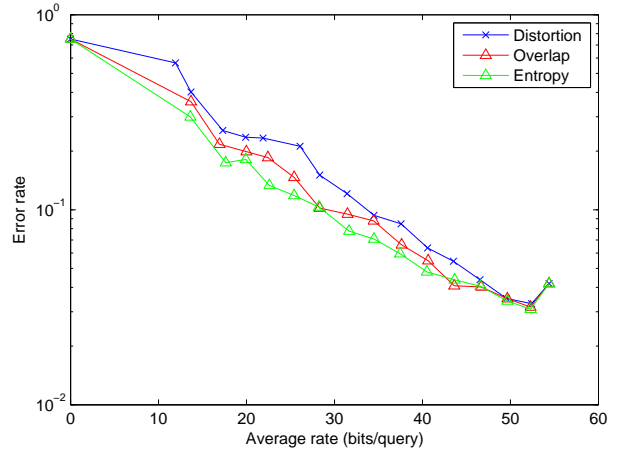


Fig. 6. Comparison of three tree pruning criteria.

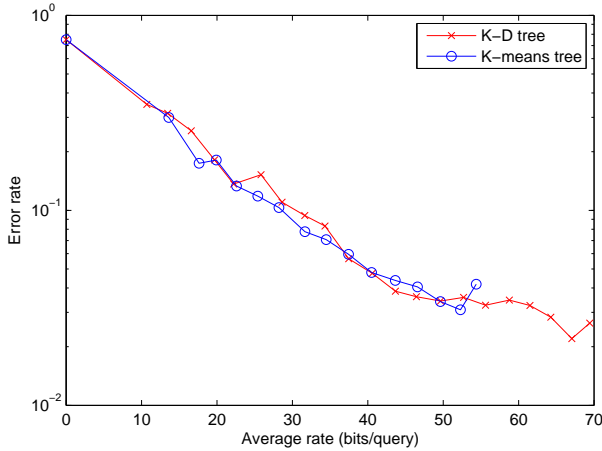


Fig. 5. Comparison of two tree building criteria.

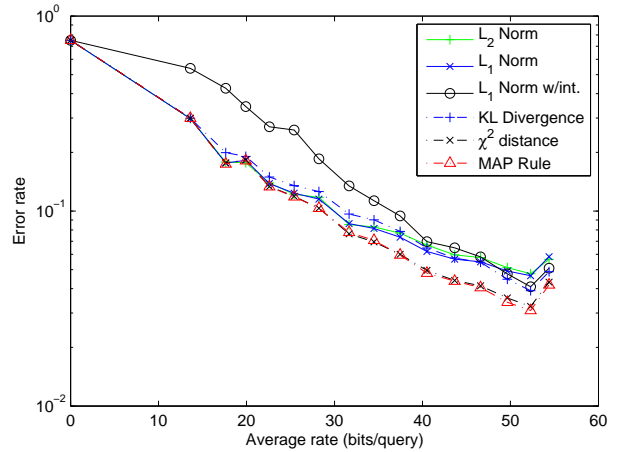


Fig. 7. Comparison of various classification criteria.

In Fig. 6, we compare the different BFOS pruning metrics for a TSVQ. We observe that the distortion-based pruning performs worse than the overlap and cluster entropy metrics. The latter two metrics perform better, as the BFOS algorithm successfully prunes away clusters that are not distinctive (i.e. that have high cluster entropy).

Last, in Fig. 7, we compare the different classification criteria discussed in Section 4.3. We observe that the L_1 -norm and L_2 -norm have similar performance, with the L_1 -norm performing slightly better. The L_1 -norm scheme that considers interior nodes performs better than schemes that ignore the interior nodes. The symmetric KL-divergence performs well at high-rates. The χ^2 distance and MAP rule out-perform other distance metrics.

6. CONCLUSIONS

In this project, we have reviewed the content-based image retrieval framework with an additional rate constraint. We have explored how to apply BFOS to optimally prune the classification tree to achieve a given rate. We devised two metrics for tree pruning: cluster entropy and cluster overlap, and compared them to distortion-based pruning. We have shown that the two different pruning metrics provide gains over using distortion as the pruning metric. Furthermore, we experimented with different classification criteria, and analyze the classification error probability. Classification based on a MAP rule performed the best in our experiments. The approach presented here appears promising, and we would like to extend our work further to real world data.

7. ACKNOWLEDGMENT

We would like to thank Kivanc Ozonat for interesting and helpful discussions during the development of this work. We also had interesting discussions with David Chen, and Gabriel Takacs along the way.

8. WORK DIVISION

1/3 - Vijay Chandrasekhar. Implemented BFOS algorithm.
1/3 - Zhi Li. Analysis. Experiments with different tree pruning criteria.
1/3 - Shang-hsuan Tsai. Experiments with different classification criteria, different tree building criteria.

9. REFERENCES

- [1] A. W. M. Smeulders, M. Worring, S. Santini, A. Gupta, and R. Jain, "Content-Based Image Retrieval at the End of the Early Years," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 22, no. 12, pp. 1349–1380, 2000.
- [2] J. Z. Wang, J. Li, and G. Wiederhold, "SIMPLIcity: Semantics-Sensitive Integrated Matching for Picture Libraries," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 23, no. 9, pp. 947–963, 2001.
- [3] D. Lowe, "Distinctive Image Features from Scale-Invariant Keypoints," *International Journal of Computer Vision*, vol. 60, no. 2, pp. 91–110, 2004.
- [4] K. Mikolajczyk and C. Schmid, "Performance Evaluation of Local Descriptors," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 27, no. 10, pp. 1615–1630, 2005.
- [5] H. Bay, T. Tuytelaars, and L. Van Gool, "SURF: Speeded Up Robust Features," in *ECCV (1)*, 2006, pp. 404–417.
- [6] K. Grauman and T. Darrell, "The Pyramid Match Kernel: Discriminative Classification with Sets of Image Features," in *ICCV '05: Proceedings of the Tenth IEEE International Conference on Computer Vision*, Washington, DC, USA, 2005, pp. 1458–1465, IEEE Computer Society.
- [7] D. Nistér and H. Stewénius, "Scalable Recognition with a Vocabulary Tree," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2006, vol. 2, pp. 2161–2168.
- [8] J. Philbin, O. Chum, M. Isard, J. Sivic, and A. Zisserman, "Object Retrieval with Large Vocabularies and Fast Spatial Matching," in *Proc. of Conference on Computer Vision and Pattern Recognition (CVPR)*, 2007.
- [9] P.A. Chou, T. Lookabaugh, and R.M. Gray, "Optimal pruning with applications to tree-structured source coding and modeling," *Information Theory, IEEE Transactions on*, vol. 35, no. 2, pp. 299–315, Mar 1989.
- [10] L Breiman, J H Friedman, R A Olshen, and C J Stone, "Classification and regression trees. the wadsworth statistics/probability series," 1984.
- [11] Jianguo Zhang, M. Marszalek, S. Lazebnik, and C. Schmid, "Local features and kernels for classification of texture and object categories: A comprehensive study," *Computer Vision and Pattern Recognition Workshop, 2006 Conference on*, pp. 13–13, June 2006.