Tiling and adaptive image compression

Wee Sun Lee*†, *Member, IEEE*

July 3, 2000

Index Terms: *Image Compression, Sequential Data Compression, Probability Modeling, Universal Source Coding, Tracking Non-stationary Source, Specialist Model, Two Dimensional Source Coding*

Abstract

We investigate the task of compressing an image by using different probability models for compressing different regions of the image. In this task, using a larger number of regions would result in better compression, but would also require more bits for describing the regions and the probability models used in the regions. We discuss using quadtree methods for performing the compression. We introduce a class of probability models for images, the k*-rectangular tilings* of an image, that is formed by partitioning the image into k rectangular regions and generating the coefficients within each region by using a probability model selected from a finite class of N probability models. For an image of size $n \times n$, we give a sequential probability assignment algorithm that codes the image with a code length which is within $O(k \log \frac{Nn}{k})$ of the code length produced by the *best* probability model in the class. The algorithm has a computational complexity of $O(Nn^3)$. An interesting subclass of the class of k-rectangular tilings is the class of tilings using rectangles whose widths are powers of two. This class is far more flexible than quadtrees and yet has a sequential probability assignment algorithm that produces a code length that is within $O(k \log \frac{Nn}{k})$ of the best model in the class with a computational complexity of $O(Nn^2 \log n)$ (similar to the computational complexity of sequential probability assignment using quadtrees). We also consider progressive transmission of the coefficients of the image.

*This work was supported in part by the National University of Singapore Academic Research Fund grant RP3992710. The material in this paper was presented in part at the IEEE International Symposium on Information Theory, Sorrento, Italy, June 25-30, 2000.

†The author is with the Department of Computer Science, School of Computing, National University of Singapore, Singapore 117543, Republic of Singapore. Email: leews@comp.nus.edu.sg.

1 Introduction

Consider the task of compressing a wavelet subband comprising $n \times n$ wavelet coefficients that have been quantized using a scalar quantizer. For natural images, it is well known that the wavelet coefficients are small in smooth areas and large in the neighbourhood of edges. Because of that, we would like to use different probability models for coding different parts of the subband in order to obtain good compression. We will restrict ourselves to a finite number N of different probability models to choose from. For example, the N models may come from the class of Generalized Gaussian Distributions; distributions in this class have been shown to be good models for compressing wavelet coefficients [10].

Under these conditions, the central question is how to partition the wavelet subband into different regions, where the coefficients in each region are coded with one of the N probability models, in order to achieve good compression performance. A trade-off is immediately obvious here. Increasing the number of different regions will allow more flexibility to fit the probability models to the coefficients. However, more regions also require a larger overhead in describing the regions and the probability model in each region.

For practical compression schemes, an important additional requirement is that the computational complexity of the method also be reasonably low. In this paper, we will discuss using quadtree methods for performing this task. Then we will examine a new method that aims to compress as well the best model in the class of probability models formed by partitioning the image into k rectangular regions and generating the coefficients within each region by using a probability model from the finite class of N probability models. We call the class of probability models that is generated in this way the class of *k-rectangular tilings* of the image.

The class of k-rectangular tilings can be considered as a natural extension to two dimensions of the class of piecewise-identically-distributed source for sequences studied in information theory [19, 11]. The coding methods for the class of piecewise-identically-distributed source have been applied to text compression to obtain some of the best text compression results currently available [16]. Similar methods have also been studied in computational learning theory [7, 17, 4]. In fact, the method described in this paper is an extension of the specialist method in [4] to two dimensions.

In this paper, we provide a sequential probability assignment algorithm that codes the image with a code length that is within $O(k \log \frac{Nn}{k})$ bits of the code length produced by the best model in the class of k-rectangular tilings of the image, where k does not need to be known in advance. We call the algorithm the *best tiling* algorithm. The computational complexity of the algorithm is $O(Nn^3)$. If we restrict the class of probability models to those generated using rectangular partitions of D discrete widths, the computational complexity can be improved to $O(Nn^2D)$. This means that we can have a fast algorithm of computational complexity $O(Nn^2W)$ for a probability assignment that is competitive with the best assignment provided by the class of k-rectangular tilings using rectangles of widths less than W. Another interesting class of models under the restriction to D discrete widths is the class of k-rectangular tilings with rectangles whose widths are powers of two. Restriction of the probability models to this class allows us to have an algorithm with a computational complexity of $O(Nn^2 \log n)$. This class is similar to the class of quadtrees but is more powerful since only one dimension is restricted to the $\log_2 n$ discrete sizes and arbitrary shifts are allowed.

The best tiling algorithm can also be used to provide progressive transmission of an image with the same bound on the code length and a computational complexity of $O(LNn^3)$ where $L-1$ is the number of refinement layers transmitted. Progressive transmission of an image is a property that is currently deemed to be practically important in applications such as browsing the world wide web and providing unequal error protection to the compressed image bitstream [13].

Many practical wavelet image compression algorithms are currently either based on trees [14, 15] or contexts [10, 2, 20]. In tree based methods, regions in an image are associated with nodes in a tree such that the regions associated with the children of a node partition the region associated with the parent node. A good pruning of the tree will provide a partition of the image with a small number of regions, where each region can be well compressed using a simple model. Some of the current wavelet image compression methods [14, 15] use trees with regions that span more than one wavelet subband. These trees, called zerotrees, are used in conjunction with progressive bit-plane by bit-plane transmission to indicate whether a coefficient is larger than a certain threshold. If all the coefficients in a node are smaller than the threshold, the children of the node can be pruned away, hence reducing the number of regions that need to be described. Transmitting the image one bit-plane at a time allows the quality of the image to progressively improve as more bits are received. Another advantage in progressive bit-plane by bit-plane transmission lies in the fact that the variance of the wavelet coefficients at the same spatial location tends to decrease as we go from a coarse subband to a fine subband along the same orientation. By using a tree that has coarse subband coefficients located only in nodes that are near the root of the tree, most of the nodes will be pruned away when the threshold used with the zerotree is large. The ordering in the variance of the coefficients can thus be exploited to improve compression.

In this paper, we will only consider the simple quadtree structure, where each rectangular region is recursively partitioned into four predefined rectangular regions. Such a simple structure is already quite effective in practice [9]. (In fact, the results in [9] using simple quadtrees are better than the results in Shapiro's original work [15].) We note also that the restriction of Shapiro's zerotree [15] to coefficients from a single subband is a quadtree. The pruning of the quadtree that minimizes the total description length can be obtained with a computational complexity of $O(n^2)$ using dynamic programming techniques. In this paper, we show how tree weighting methods [18, 6] can be used to provide sequential probability assignment using the quadtree. The weighting method has a higher computational complexity of $O(n^2 \log n)$. However, it can be shown to compress at least as well as (and usually better than) the optimal pruning method if a single scalar quantizer and the same prior are used in both methods. Another advantage of the tree weighting method, when used with a single scalar quantizer, comes from the fact that progressive transmission obtains the same code length as that obtained by non-progressive transmission. The tree weighting method can be readily generalized from quadtrees to other tree structures.

In context based methods [10, 2, 20], the coefficients that have been already been transmitted are used to select the current state of the encoder. A probability model associated with the state is then used to code the coefficient. As described, the context based method is very general and covers the tiling and tree weighting methods. One method to capture the intuition that similar coefficients tend to occur together in an image is to use only transmitted coefficients in a small window around the coefficient currently being coded to select the state of the encoder. This is done in [10], where the maximum likelihood distribution found by using only the coefficients inside the window is used to code the coefficient. The context based method can also exploit information from transmitted coefficients in other subbands by using them in the mapping to the state of the encoder [2, 20]. By using clever mappings that exploit properties of the image, coding gains can be achieved [20].

Different context based algorithms may perform well in different regions of the image. We note that the finite set of N probability models used by the quadtree and tiling methods can contain sequential probability assignment algorithms instead of static probability models. Hence, context based methods can be used in conjunction with the quadtree weighting method or the best tiling method by being included in the set of N probability models. This is analogous to the coding methods used in [16] which switch between two coding algorithms with different properties.

The main contribution of the paper is the introduction of the class of k-rectangular tilings of an image as a useful comparison class for image compression and showing that a computationally efficient compression method that performs almost as well as the best model in the class exists. In Section 2, we give some definitions and describe some of the basic concepts used in this paper. The quadtree coding methods are described in Section 3. In Section 4, we develop the probability assignment method that is competitive with the class of k-rectangular tilings of an image and consider more restricted variants that are computationally simpler. Progressive transmission using the method is also discussed.

2 Preliminaries

An image is a matrix of coefficients with a finite number rows and columns. For convenience, we will often refer to an image as a sequence of coefficients when the two dimensional characteristics of the image is not important.

2.1 Quantization.

In *vector quantization*, an m-dimensional vector of coefficients at time t, $\mathbf{x}_t = (x_{tm}, x_{tm+1}, \ldots, x_{(t+1)m-1}) \in \Re^m$, is mapped to a reproduction vector $\mathbf{u}_t = (u_{tm}, u_{tm+1}, \ldots, u_{(t+1)m-1}) \in \mathcal{U}$, where $\mathcal{U}$ is a finite set of reproduction vectors. The mapping from $\mathbf{x}_t$ to $\mathbf{u}_t$ is called a *vector quantizer*. A vector quantizer with $m = 1$ is called a *scalar quantizer*. One way to define a scalar quantizer is to use a sequence of thresholds $-\infty \leq b_0 \leq b_1 \leq \cdots \leq b_M < \infty$ and a sequence of reproduction values $-\infty \leq r_0 \leq r_1 \leq \cdots \leq r_M < \infty$. When the thresholds $b_0, \ldots, b_M$ are equally spaced, we call the quantizer a *uniform quantizer*. In a uniform quantizer with a *deadband*, the size of the quantization intervals $b_i - b_{i-1}$ are the same except for the interval containing zero which is twice as large as the other intervals.

To obtain the optimal code length at a fixed distortion, it is necessary to use vector quantization. However, the computational complexity of using vector quantization can be considerable. One method for handling the complexity while achieving good performance is to use simple scalar quantization together with good entropy coding of the quantized coefficients. This can be shown to be within 0.255 bits per coefficient of optimal for high rates and within 0.754 bits per coefficient of optimal at low rates (see [5] for details). In practice, linear transforms that compact energy provide coding gain. This coding gain can be easily exploited in the transform domain using scalar quantization. State of the art coders such as those in [10, 2] use different scalar quantizers for different regions of the wavelet subband while the coders in [15, 14, 20] use a single scalar quantizer with a deadband for all the coefficients.

In this paper, we will use a single scalar quantizer for all the coefficients. The range of the scalar quantizer is a finite set of values. For simplicity, we will assume that the range is $\{0, \ldots, b-1\}$ when the alphabet size is b. Let the probability assigned to a scalar quantized image $\{x_0, \ldots, x_{t-1}\}$ be $p(x_0, \ldots, x_{t-1})$. Using an arithmetic coder, the *ideal* code length $\log_2 1/p(x_0, \ldots, x_{t-1})$ can be approached very closely. Throughout this paper, ideal code lengths which can be closely approximated using arithmetic coding will be used.

2.2 Models for Images

Let C be a finite class of probability distributions for coefficients quantized using a scalar quantizer. Partition the image into a number of subsets called *regions*. Each region R is assigned a probability distribution from C, which is then used for generating all the coefficients in R. In this way, *models for images* are generated from models for coefficients and image partitions.

2.3 Adaptation

Let F be a finite class of models for images. We put a prior probability distribution $p(f)$ on F. The class F will serve as a *benchmark* for judging how well we perform in our task of coding the image. Our aim is to code the image in such a way that the resulting code length is close to the best that can be achieved by using the best model in F.

Let $x^t = \{x_0, \ldots, x_{t-1}\}$ represent the image to be coded after scalar quantization with a single quantizer. Let f_{opt} be a model that produces the shortest code length when used to code x^t. The ideal code length produced by f_{opt} is $\log_2 1/p(x^t|f_{\text{opt}})$. Let c_A be the code length produced by the algorithm A when it is used to code x^t. The *redundancy* of algorithm A for coding x^t with respect to the class F is $c_A - \log_2 1/p(x^t|f_{\text{opt}})$.

2.3.1 Forward Adaptation

Conceptually, the simplest method for adaptive coding is the forward adaptation method. In the forward adaptation method, the model $f \in F$ that minimizes the *total description length* $\log_2 1/p(x^t|f) + \log_2 1/p(f)$ is first found. Let the model be f'_{opt}. The description of f'_{opt} is then transmitted using $\log_2 1/p(f'_{\text{opt}})$ bits. This is followed by the description of the coefficients of length $\log_2 1/p(x^t|f'_{\text{opt}})$. Note that the model f'_{opt} is not necessarily the same as f_{opt} since it is the model that optimizes the total description length and not the model that optimizes the description length of the coefficients only. From the definition of f'_{opt},

$$\log_2 1/p(x^t|f'_{\mathrm{opt}}) + \log_2 1/p(f'_{\mathrm{opt}}) \le \log_2 1/p(x^t|f_{\mathrm{opt}}) + \log_2 1/p(f_{\mathrm{opt}}).$$

Hence, the redundancy of the forward adaptation method is bounded by $\log_2 1/p(f_{\mathrm{opt}})$. If the uniform prior is put on F, the redundancy is bounded by $\log_2 |F|$.

2.3.2 Backward Adaptation

From the chain rule of probability, we can write

$$p(x_0, \ldots, x_{t-1}) = p(x_0)p(x_1|x_0)\cdots p(x_i|x^i)\cdots p(x_{t-1}|x^{t-1}).$$

This means that by coding the image sequentially using the conditional probability $p(x_i|x^i)$, we can achieve the code length $\log_2 1/p(x_0, \ldots, x_{t-1})$. With a model class F of size M and prior distribution $p(f)$, we have

$$p(x_0, \ldots, x_{t-1}) = \sum_{i=0}^{M-1} p(f_i)p(x_0, \ldots, x_{t-1}|f_i) \ge p(f'_{\mathrm{opt}})p(x_0, \ldots, x_{t-1}|f'_{\mathrm{opt}}).$$

This means that backward adaptation with the conditional probability will perform at least as well as (and most likely better than) the forward adaptation method using the same model class and prior distribution.

We call an algorithm that produces $p(x_i|x^i)$ for a class of models F and a prior distribution $p(f)$ a *Bayesian sequential probability assignment* method. One interesting property of the Bayesian sequential probability assignment method is that the code length produced is invariant to the ordering of the coefficients, as can be seen from the form of the chain rule of probability. Sequential updating for Bayesian sequential probability assignment is relatively simple. We have

$$\begin{aligned} p(x_t|x^t) &= \frac{p(x_0, \ldots, x_t)}{p(x_0)p(x_1|x_0)\cdots p(x_{t-1}|x^{t-1})} \\ &= \frac{\sum_{i=0}^{M-1} p(f_i)p(x_0, \ldots, x_t|f_i)}{p(x_0)p(x_1|x_0)\cdots p(x_{t-1}|x^{t-1})} \\ &= \frac{\sum_{i=0}^{M-1} p(f_i)p(x_0, \ldots, x_{t-1}|f_i)p(x_t|f_i)}{p(x_0)p(x_1|x_0)\cdots p(x_{t-1}|x^{t-1})}. \end{aligned} \tag{1}$$

Assume that model i is initialized to prior probability p_0^i. Let $a_x^i = p(X = x|f_i)$ for $x = 0, \ldots, b-1$, where b is the alphabet size. Let $\mathbf{a}^i = (a_0^i, \ldots, a_{b-1}^i)$ be the probability vector associated with model i. The Bayesian sequential probability assignment algorithm is can now be rewritten as follows:

1. Predict with the weighted average of the predictions of the models:

$$\mathbf{a}_t = \sum_{i=0}^{M-1} p_t^i \mathbf{a}^i.$$

2. Observe the outcome x_t.

3. Calculate the a posteriori probability:

$$p_{t+1}^i = \frac{p_t^i a_{x_t}^i}{a_{t,x_t}}.$$

Backward adaptation need not be restricted to using conditional distributions derived from a function class F and a prior distribution on F. Any sequential probability assignment function that produces a conditional probability distribution $a(x_i|x^i)$ can be used. If the sequential probability assignment function assigns a high probability to the image, a short code length will be produced. A good survey of sequential probability assignment can be found in [12].

2.4 Progressive Transmission

A scalar quantizer Q maps all the real-valued coefficients from an interval to one of a finite number of values. We say that a scalar quantizer Q_i is a *refinement* of a scalar quantizer Q_{i-1} if for every a in the range of Q_i, $Q_i^{-1}(a) \subseteq Q_{i-1}^{-1}(b)$ for some b in the range of Q_{i-1}. In other words, each interval corresponding to Q_i is a subset of some interval corresponding to Q_{i-1}.

Let $x_{-1}^t = \{x_{0,-1}, \ldots, x_{t-1,-1}\}$ be a the image produced by quantizer Q_{-1} and $x^t = \{x_0, \ldots, x_{t-1}\}$ be the image produced by quantizer Q, which is a refinement of quantizer Q_{-1}. Then

$$\begin{aligned} p(x_0, \ldots, x_{t-1}) &= p(x_0, \ldots, x_{t-1}, x_{0,-1}, \ldots, x_{t-1,-1}) \\ &= p(x_0, \ldots, x_{t-1}|x_{0,-1}, \ldots, x_{t-1,-1})p(x_{0,-1}, \ldots, x_{t-1,-1}). \end{aligned}$$

This means that we can achieve the code length $\log_2 1/p(x_0, \ldots, x_{t-1})$ in two steps. First we quantize the image using a coarse quantizer Q_{-1} and transmit it using the Bayesian sequential probability assignment method. Next, we quantize the image using Q, which is a refinement of Q_{-1}, and transmit it using the Bayesian sequential probability assignment method conditioned upon the value of the coarse version.

The Bayesian sequential probability assignment method with scalar quantization achieves the same code length regardless of whether progressive transmission is used or not. However, it does not achieve the *optimal rate-distortion* representation at each stage. That requires the rate distortion problem to be successively refinable [3] and is unlikely to be achievable using a scalar quantizer.

3 Quadtree Methods

To specify a quadtree, we start with a rectangular image. The image is associated with the *root* of the quadtree. The image is partitioned into four subsets called *regions*. In practice, the regions are usually rectangular. These four regions are associated with four nodes that are the *children* of the root. The children are connected to their *parent* (the root) by edges. Each edge is labeled by a symbol from an alphabet of four symbols. If desired, each node can be further partitioned into four children in the same way as the root. The process can be continued as long as each node is nonempty. Nodes that are not partitioned further are called *leaves*. Nodes that are not leaves are called *internal nodes*. Each leaf can be assigned a probability model for coefficients from a finite class C of N probability models. The probability model is used to code all the coefficients in the leaf. Each node can be uniquely identified by a path from the root to the node. We call the structure defined this way a quadtree. Each quadtree defines a set a probability models for the image where each assignment of probability models from C to the leaves of the quadtree defines a *quadtree model* for the image.

A *template* quadtree is a quadtree that is known to both the encoder and the decoder. A *pruning* P of a template quadtree T is a quadtree induced by replacing some internal nodes of T with a leaves.

3.1 Forward Adaptation

Given a template quadtree T, we now describe how to code the description of a quadtree model from the pruned quadtree P. Start the coding procedure with the root. If the current node is a leaf of T, transmit the description of the model associated with the leaf. If the current node is a leaf of P but not of T, transmit the symbol zero followed by the description of the model associated with the leaf. The additional bit is not needed if the node is a leaf of T since the decoder knows that the node has to be a leaf. If the current node is not a leaf, transmit the symbol one. Then recursively code the four children in a fixed order using the procedure just described. Using this coding procedure, each quadtree model is assigned a description from a prefix code. The procedure also assigns a prior probability distribution to the class of quadtree models formed from all possible prunings of T and assignments of models to the leaves of the prunings. The prior probability assigned to a model with a code length l is 2^{-l}.

We now describe a simple dynamic programming algorithm for finding the pruned quadtree model that minimizes the total description length of transmitting the image. Start the routine *Prune* by using the root as a parameter. If the current node is a leaf, the routine returns the minimum total description length of coding the node as the *cost* of the node. If the current node is an internal node, *Prune* calls itself four times with the four children as the parameters. It then compares the following two sums: the minimum total description length of coding the node plus one and the sum of the costs of all its children. If the former is smaller, the children of the nodes are pruned away and the former becomes the cost of the node. Otherwise, the children are retained and the latter becomes the cost of the node. One bit is kept at the node to indicate whether or not the children are pruned away. The routine then returns the cost of the node.

The number of leaves in the template quadtree is at most n^2 for an n by n image. Hence, the total number of nodes is at most $4n^2/3$. This means that the computational complexity of the algorithm is $O(Nn^2)$.

3.2 Backward Adaptation

We want to calculate $p(x_0, \ldots, x_{t-1}) = \sum_{i=0}^{M-1} p(f_i)p(x_0, \ldots, x_{t-1}|f_i)$ using the Bayesian sequential probability assignment method. The M models in the class include the models from all prunings of the template tree. The prior probability is the probability assigned by the coding process described in Section 3.1. Since M is very large, the direct method of performing the Bayesian sequential probability assignment method described in Section 2.3.2 is highly inefficient.

The tree weighting algorithm used here is a variant of the weighting algorithms presented in [6, 18]. The algorithm uses the whole template tree T. The models in each node are initialized to the prior distribution on C. At time t, the coefficient traces a path from the root to the leaf; hence we can partition the nodes in T into those that are on the path and those that are not on the path. Let $\mathbf{a}_t$ be the probability vector produced by the algorithm for coding the coefficient at time t and let $\mathbf{a}^j = (a_0, \ldots, a_{b-1})$ be the probability vector associated with model $c_j \in C$ with an alphabet of size b. We can rewrite equation (1) as follows

$$\mathbf{a}_t = \sum_{j=1}^{N} W^t_{\text{total}}(r, j)\mathbf{a}^j$$

where r is the root of T. The function $W^t_{\text{total}}(u, j)$ is the sum of the weights of all the models generated by all possible prunings of the subtree of T rooted at u that have c_j as the path leaves. This function is calculated by

$$W^t_{\text{total}}(u,j) = \begin{cases} W^t_{\text{model}}(u,j) & \text{path leaf} \\ \\ \frac{1}{2}W^t_{\text{model}}(u,j) & u \text{ path internal node,} \\ +\frac{1}{2}W^t_{\text{total}}(s,j)\prod_{v\in\text{children of } u, v\neq s} W^t_{\text{node}}(v) & s \text{ path node child of } u \end{cases}$$

where the the functions $W^t_{\text{model}}(u,j)$ and $W^t_{\text{node}}(u)$ are updated by

$$W^{t+1}_{\text{model}}(u,j) = \begin{cases} W^t_{\text{model}}(u,j) & \text{off path} \\ \\ W^t_{\text{model}}(u,j)a^j_{x_t}/a_{t,x_t} & \text{on path} \end{cases}$$

and

$$W^{t+1}_{\text{node}}(u) = \begin{cases} W^t_{\text{node}}(u) & \text{off path} \\ \\ \sum_{j=1}^N W^{t+1}_{\text{model}}(u,j) & \text{path leaf} \\ \\ \frac{1}{2}\sum_{j=1}^N W^{t+1}_{\text{model}}(u,j) + \frac{1}{2}\prod_{v\in\text{children of } u} W^{t+1}_{\text{node}}(v) & \text{path internal node.} \end{cases}$$

All the calculations happen only along the path of the coefficient and hence the computational complexity for each coefficient is proportional to the length of the path. If each node is partitioned into four (approximately) equal sized children, the height of the tree is $O(\log n)$. Hence the total computational complexity is $O(n^2 \log n)$.

We now prove that the weights given by the algorithm are correct.

Theorem 1

$$\mathbf{a}_t = \sum_{j=1}^N W^t_{total}(r,j)\mathbf{a}^j = \frac{\sum_{i=0}^{M-1} p(f_i)p(x_0,\ldots,x_{t-1}|f_i)p(x_t|f_i)}{p(x_0)p(x_1|x_0)\cdots p(x_{t-1}|x^{t-1})}$$

where r is the root of T.

Proof. If T consist of a single node, the result follows from Section 2.3.2. We now show that $W^t_{\text{total}}(u,j)$ is calculated correctly for node u when u has children.

Note that the description length of a model can be decomposed into the sum of the description length of the tree structure and the description lengths of all the models at the leaves. Denote the description length of a tree P as $|P|$. Let the description length of a model f be $|P|+L_f$. Hence the prior probability of f is $2^{-|P|-L_f}$. The posterior probability of f is $2^{-|P|}w(f)$ where

$$w(f) = \frac{2^{-L_f}p(x_0|f)p(x_1|f)\cdots p(x_{t-1}|f)}{p(x_0)p(x_1|x_0)\cdots p(x_{t-1}|x^{t-1})}, \tag{2}$$

assuming that the coefficients are independent given f. Any pruning P of the subtree T_u rooted at u contains either only the node u or can be decomposed into four subtrees P_0, P_1, P_2, P_3 rooted at the four children of u. From the way the description length is constructed, we have $|P| = 1 + |P_0| + |P_1| + |P_2| + |P_3|$. Let f^{P_i} be the function f restricted to the subset associated with P_i. The description length L_f can be written as $L_{f^{P_0}} + L_{f^{P_1}} + L_{f^{P_2}} + L_{f^{P_3}}$. We can also rewrite equation (2) as

$$w(f) = \prod_{i' \in \{0,1,2,3\}} 2^{-L_{f^{P_{i'}}}} w(f^{P_{i'}})$$

where $w(f^{P_i}) = \prod_m \frac{p(x_m|f)}{p(x_m|x^m)}$ for all x_m that lie in the subset associated with P_i. Note that only one of the four children is on the path of the coefficient. Without loss of generality, let that particular subtree be P_0.

If u does not have children, $W^t_{\text{total}}(u,j) = W^t_{\text{model}}(u,j)$. Hence we have

$$\begin{aligned} W^t_{\text{total}}(u,j) &= 2^{-1}W^t_{\text{model}}(u,j) + \\ &\quad \sum_{P_0,f^{P_0}} \sum_{P_1,f^{P_1}} \sum_{P_2,f^{P_2}} \sum_{P_3,f^{P_3}} 2^{-(1+|P_0|+|P_1|+|P_2|+|P_3|)} \prod_{i' \in \{0,1,2,3\}} 2^{-L_{f^{P_{i'}}}} w(f^{P_{i'}}) \\ &= \frac{1}{2}W^t_{\text{model}}(u,j) + \frac{1}{2} \prod_{i' \in \{0,1,2,3\}} \sum_{P_0,f^{P_{i'}}} 2^{-|P_{i'}|} 2^{-L_{f^{P_{i'}}}} w(f^{P_{i'}}) \end{aligned}$$

where the functions f^{P_0} that are assigned are restricted so that all leaves in the path of the current coefficient are restricted to have the model c_j.

The term $\sum_{P_0,f^{P_0}} 2^{-|P_0|}2^{-L_{f^{P_0}}} w(f^{P_0})$ is just $W^t_{\text{total}}(v,j)$ where v is the root of P_0. The other terms are denoted $W^t_{\text{node}}(v)$, where the v takes on the value of the roots of P_1, P_2 and P_3. The calculation of $W^t_{\text{node}}(v)$ is the same as the calculation of $W^t_{\text{total}}(v,j)$, except that no restriction to c_j is done since all the nodes are not in the path of the current coefficient. □

Progressive transmission is done the same way as non-progressive transmission, except that during the refinement stage i, $p(x_{t,i}|c, x_{t,i-1})$ is used in place of $p(x_t|c)$ for $c \in C$.

The template quadtree provides $O(n^2)$ rectangles which are in fixed positions. Because the rectangles are in fixed positions, the performance of any algorithm based on the quadtree is sensitive to shifts in the underlying regions of the image. Regions that are poorly placed with respect to the quadtree can only be well described using a pruned quadtree with many leaves. Consequently the overhead cost for describing such regions will be higher. This motivates the use of the class of k-rectangular tilings of the image to be described next.

4 Rectangular tiling of an image

In this paper, we introduce a class of probability model for images: the *k-rectangular tilings* of an image. This class is formed by partitioning the image into k rectangular regions and generating the coefficients within each region by using a probability model from a finite class C of probability models for coefficients.

We first consider the redundancy of the Bayesian sequential probability assignment on this class. Since there are n^2 pixels in the image, there are $n^2 + n^2(n-1) + (n^2(n^2-1)/2 - n^2(n-1))/2 = n^2 + n^2(n-1)/2 + n^2(n^2-1)/4$ distinct rectangular regions; the first term, n^2, counts the number of rectangles that contain a single coefficient; the second term, $n^2(n-1)$, counts the number of rectangles that have either a single row or a single column; the third term, $(n^2(n^2-1)/2 - n^2(n-1))/2$, counts the number of rectangles that have more than one row and more than one column. Each region can be assigned one of the N probability models in C. If there are k regions in the image, the number of probability models in the class is upper bounded by $N^k n^{4k}$. Assuming a uniform prior is used, the Bayesian sequential probability assignment method gives a redundancy of at most $k \log_2 N + 4k \log_2 n$.

Unfortunately, unless some special structure is exploited, the Bayesian probability assignment method suffers from computational intractability when the number of models is large. It is easy to see that even if we fix the structure of the tiling, there are N^k different models in the class, each of which needs to be updated each time a coefficient is encoded.

The tree weighting method described in Section 3 exploits the tree structure to perform sequential Bayesian probability assignment in a computationally efficient manner. When the data forms a sequence, and not an image, computationally efficient methods also exist for performing Bayesian weighting on models formed by segmenting the sequence into segments and using a different probability model to generate the coefficients in each segment [7, 17]. The simpler versions of these algorithms can be considered as certain Hidden Markov models that switch from the current model to another model with a certain probability.

Unfortunately, we have not been able to extend the methods in [7, 17] from segmentation of a sequence to tiling of a plane. The lack of causality in two dimensions appears to be the main difficulty in finding computationally efficient Hidden Markov type models. Instead, we will use a sequential probability assignment method called SBayes (Bayes for specialists) by Freund, Schapire, Singer and Warmuth [4]. SBayes is an adaptation of the Bayes method for models that are allowed to abstain. SBayes is no longer strictly a Bayesian method but it retains some useful properties similar to that of Bayesian methods on redundancy while at the same time providing some attractive computational properties.

4.1 Specialist Bayes

The *specialist framework* was first proposed by Blum [1], as an extension of the online prediction by experts framework commonly studied in computational learning theory. In the online prediction by experts framework, the cumulative loss of an online prediction algorithm is compared against the cumulative loss of the best predictor in a class of predictors (experts). In this framework, the Bayesian sequential probability allocation algorithm can be shown to have a redundancy of $O(\log N)$ relative to a class of N experts when the log loss is used. Similar bounds can also be given for related algorithms for other loss functions such as the absolute loss and the squared loss.

In the specialist framework, the experts are allowed to abstain and are called specialists in analogy to human specialists who make predictions only when the instance to be predicted falls within their area of specialty. In this paper, we will call a specialist that predicts using a fixed probability model and abstains outside its region of expertise, a *specialist model*. Whenever a specialist model is not abstaining we say that it is *active* and we call the region where the specialist is active its *region of activity*. We call a subset of specialists whose regions of activity partition the sequence of symbols an *extended model*. For an extended model, there is always only one model that will make a prediction for every member of the sequence. As such, the extended model forms a probability model for sequences (or for images if the sequence consist of all the coefficients in an image). In this paper, we assume that a subset of specialists can always be found to form at least one extended model. This avoids the problem of having instances on which all specialist models abstain. By careful construction of the class of specialist models, we can always ensure that this assumption is satisfied.

For this paper, we will assume a finite set S of specialists and an alphabet of size b. The aim of our algorithm is to sequentially assign a probability mass function $a(x_0)a(x_1|x_0)\cdots a(x_i|x^i)\cdots a(x_{t-1}|x^{t-1})$ to a sequence of symbols $x_0, \ldots, x_{t-1}$. At iteration i, the algorithm needs to produce $a(x_i|x^i)$. We perform our analysis in a deterministic setting, where the target symbol x_i can be chosen by an adversary with knowledge of the probability assignment algorithm we are using and the assignments that we have made up to and including time i. The output $a(\cdot|x^i)$ of a sequential probability assignment algorithm is a probability vector $\mathbf{a_i} = (a_{i,0}, \ldots, a_{i,b-1})$. The algorithm suffers a loss of $\log_2 1/a_{i,x_i}$ when the outcome is x_i at iteration i. The cumulative sum of the losses of algorithm is the code length assigned by the algorithm to the sequence $-\sum_{i=0}^{t-1} \log_2 a(x_i|x^i)$.

Let F be a set of extended models that are formed from the the class of specialist models S. We would like to bound the redundancy of the algorithm relative to the class F.

In the SBayes algorithm, each specialist model is given a weight that summarizes the contributions of the specialist model so far. These weights play the same role as the a posteriori probability of a model in the normal Bayesian algorithm. Initially, the weights of the specialist models are initialized to a "prior" probability distribution $\mathbf{p_0} = (p_0^0, \ldots, p_0^{m-1})$, which is usually the uniform distribution. At each iteration, the set of active specialist models E_i is treated as if it were the complete set of models. On receiving the ith symbol, only the weights of active specialist models are updated while the weight of the abstaining specialist models are untouched. Assume that the specialist model j outputs $\mathbf{a}^j = (a_0^j, \ldots, a_{b-1}^j)$ and the algorithm SBayes outputs $a(\cdot|x^i) = \mathbf{a}_i = (a_{i,0}, \ldots, a_{i,b-1})$ at time i. The algorithm SBayes [4] is given below:

1. Predict with the weighted average of predictions of the active specialist models:

$$\mathbf{a}_i = \frac{\sum_{j \in E_i} p_i^j \mathbf{a}^j}{\sum_{j \in E_i} p_i^j}$$

2. Observe the outcome x_i.

3. Calculate the new weight:

$$p_{i+1}^j = \begin{cases} \frac{p_i^j a_{x_i}^j}{a_{i,x_i}} & \text{if } j \in E_i \\ p_i^j & \text{otherwise.} \end{cases}$$

Observe the similarity between SBayes and the Bayesian sequential probability assignment algorithm described in Section 2.3.2. The following result from Freund, et al. [4] bounds the performance of SBayes, with respect to a class of extended models. (The result given by Freund, et al. [4] is actually slightly more general than the result given below.) We say that a probability vector $\mathbf{u} = (u^0, \ldots, u^{m-1})$ over the set of specialists S of size m is associated with an extended model U with $|U| = k$ if and only if $u^i = 1/k$ whenever specialist i is in the set U and $u^i = 0$ whenever specialist i is not in U. For two probability vectors, $\mathbf{u}$ and $\mathbf{v}$, the relative entropy, $\mathbf{RE}(u||v)$ is $\sum_i u^i \log_2(u^i/v^i)$.

Theorem 2 *Let $\mathbf{u}$ be a probability vector associated with an arbitrary extended model U in a class of specialist models S. Let $a(\cdot|x^i)$ be the output at time i of SBayes which sequentially allocates probability mass using the class S. Then*

$$-\sum_{i=0}^{t-1} \log_2 a(x_i|x^i) - (-\sum_{i=0}^{t-1} \log_2 p(x_i|x^i, U)) = |U|(\mathbf{RE}(\mathbf{u}||\mathbf{p_0}) - \mathbf{RE}(\mathbf{u}||\mathbf{p_t})),$$

where $\mathbf{p_i}$ is the probability vector over the set of specialist models S at time i. If $|U| = k$, $|S| = m$ and $\mathbf{p_0}$ is the uniform distribution, then

$$-\sum_{i=0}^{t-1} \log_2 a(x_i|x^i) - (-\sum_{i=0}^{t-1} \log_2 p(x_i|x^i, U)) \leq k \log_2 \frac{m}{k}.$$

Proof. First we note that if p_i is a probability vector, then p_{i+1} is still a probability vector after an update using SBayes. We calculate

$$\begin{aligned}
\mathbf{RE}(\mathbf{u}||\mathbf{p_i}) - \mathbf{RE}(\mathbf{u}||\mathbf{p_{i+1}}) &= \sum_{j=1}^{m} u^j \log_2 \frac{p_{i+1}^j}{p_i^j} \\
&= \sum_{j \in E_i} u^j \log_2 \frac{p_{i+1}^j}{p_i^j} \\
&= \frac{1}{|U|} \log_2 \frac{p_{i+1}^{j_i}}{p_i^{j_i}} \\
&= \frac{-\log_2 a(x_i|x^i)}{|U|} - \left(\frac{-\log_2 p(x_i|x^i, U)}{|U|} \right)
\end{aligned}$$

where j_i is the index of the active specialist in U at time i.

Summing the equality from $i = 0$ to $t - 1$ gives the first result. The second follows from the fact that relative entropy is always nonnegative. $\Box$

The result shows that the algorithm SBayes provides a probability assignment with a redundancy of no more than $k \log_2 \frac{m}{k}$ relative to the class of extended models that contain no more than k specialist models when m is the size of the class of specialist models. Note that the bound holds for all k less than or equal to m for which an extended model exists.

One useful property of Bayesian sequential probability assignment using a class of extended models is that if we permute the order in which the symbols are processed, the final probability assignment is still the same. The following example shows that this property is not always retained by the SBayes method.

Example 3 *Consider a sequence of three symbols* $(x_0 = 1, x_1 = 1, x_2 = 1)$ *over the alphabets* $\{0, 1\}$*. Let the class of specialists be* $s_0 = (0.2, *, *)$*,* $s_1 = (*, 0.2, 0.2)$*,* $s_2 = (0.8, *, *)$*,* $s_3 = (*, 0.8, 0.8)$*,* $s_4 = (0.2, 0.2, *)$*,* $s_5 = (*, *, 0.2)$*,* $s_6 = (0.8, 0.8, *)$ *and* $s_7 = (*, *, 0.8)$ *where each number represent the probability that the corresponding symbol is* 1 *and* '$*$' *denotes that the specialist model is abstaining. If the symbols are processed in the order* x_0, x_2, x_1*, the probability assigned would be 0.170. However, it the symbols are processed in the order* x_0, x_1, x_2*, the probability assigned is 0.172.*

The example also shows that in this case, SBayes is not the Bayesian probability assignment method operating on some extended model class.

4.2 Tiling with Rectangles

Using SBayes, we will be able to obtain a probability assignment that is close to the probability assignment provided by the best model in class of k-rectangular tilings of the image for arbitrary k. For an $n \times n$ image, the number of possible distinct tiles is $O(n^4)$. If we have N probability models, then the total number of specialist models is $O(Nn^4)$. This gives a redundancy of $O(k \log \frac{Nn}{k})$ when compared against the class containing k or fewer tiles.

A simple algorithm that goes through each specialist model for every coefficient of the image will require a computational complexity of $O(Nn^6)$ for assigning probability to the image. We would like to reduce the computational complexity by using careful recursive updating.

We first review the one dimensional case that was examined in [4]. (In [4], the more general case of unknown sequence length is considered. Here, we consider the case where the sequence length n is known in order to generalize the result to tiling of images.) For a sequence of length n, there are $n(n+1)/2$ possible segments giving $Nn(n+1)/2$ specialist models. Let $S(t_1, t_2, j)$ be the specialist associated with model $c_j \in C$ that is active in the interval $[t_1, t_2]$. Assume that initially, they are all given equal weights which can be set to 1 because of the normalization that is performed at each iteration. Let $p^i_{t_1,t_2,j}$ be the weight of specialist $S(t_1, t_2, j)$ at time i. Then the prediction of the algorithm at time i is

$$\mathbf{a}_i = \frac{\sum_{j=0}^{N-1} \sum_{t_1=0}^{i} \sum_{t_2=i}^{n-1} p^i_{t_1,t_2,j} \mathbf{a}^j}{\sum_{j=0}^{N-1} \sum_{t_1=0}^{i} \sum_{t_2=i}^{n-1} p^i_{t_1,t_2,j}},$$

where $\mathbf{a}^j$ is the vector of prediction of the model c_j. Letting

$$Q_j^i = \sum_{t_1=0}^{i} \sum_{t_2=i}^{n-1} p_{t_1,t_2,j}^i,$$

we can rewrite the sum as

$$\mathbf{a}_i = \frac{\sum_{j=0}^{N-1} Q_j^i \mathbf{a}^j}{\sum_{j=0}^{N-1} Q_j^i}.$$

On observing the outcome x_i, the weight of an active specialist $S(t_1, t_2, j)$ is updated by multiplication with $R_j^i = a_{x_i}^j / a_{i,x_i}$. We can then write

$$Q_j^i = \sum_{t_1=0}^{i} \sum_{t_2=i}^{n-1} p_{t_1,t_2,j}^i = \sum_{t_1=0}^{i} \sum_{t_2=i}^{n-1} \prod_{s=t_1}^{i-1} R_j^s = (n-i) \sum_{t_1=0}^{i} \prod_{s=t_1}^{i-1} R_j^s.$$

Updating Q_j^i, we obtain

$$\begin{aligned}
Q_j^{i+1} &= \sum_{t_1=0}^{i+1} \sum_{t_2=i+1}^{n-1} \prod_{s=t_1}^{i} R_j^s \\
&= \sum_{t_1=0}^{i} \sum_{t_2=i+1}^{n-1} \prod_{s=t_1}^{i} R_j^s + n - i - 1 \\
&= \sum_{t_1=0}^{i} \left(\sum_{t_2=i}^{n-1} R_j^i \prod_{s=t_1}^{i-1} R_j^s - R_j^i \prod_{s=t_1}^{i-1} R_j^s \right) + n - i - 1 \\
&= R_j^i Q_j^i - R_j^i \sum_{t_1=0}^{i} \prod_{s=t_1}^{i-1} R_j^s + n - i - 1
\end{aligned}$$

The first term updates all the current specialists. The second term subtracts all the specialist models that will no longer be active at time $i+1$ while the third term adds the new specialist that will become active at time $i+1$. This can be further simplified to

$$Q_j^{i+1} = (n-i-1) \left(\frac{R_j^i Q_j^i}{(n-i)} + 1 \right).$$

We now consider the case of tiling an $n \times n$ image with rectangles of arbitrary height but one fixed width w. We assume that the coefficients are processed in a raster scan order. The origin is at the top left hand corner of the image and the coordinates increase in the direction of the scan. Let $S((a,b),(c,d),j)$ be the specialist model associated with model c_j that is active in the rectangle with top left hand corner (a,b) and bottom right hand corner (c,d). For simplicity, we associate specialist models with all rectangles with width w that intersect the image even though part of the rectangle may be outside the image. All the specialist models are given equal weights initially. Let $R(y,z)$ be the set of rectangles that contains the coordinate (y,z). Let

$$Q_j^{(y,z)} = \sum_{m \in R(y,z)} p_{m,j}^{(y,z)}$$

be the sum of the weights of the active specialist models associated with model j and the set $R(y,z)$ when the (y,z) coefficient is being processed. So we have

$$\mathbf{a}_{(y,z)} = \frac{\sum_{j=0}^{N-1} Q_j^{(y,z)} \mathbf{a}^j}{\sum_{j=0}^{N-1} Q_j^{(y,z)}}.$$

Let $R_j^{(y,z)} = a^j_{x_{(y,z)}} / a_{(y,z),x_{(y,z)}}$. Let $M(p,q)$ be a subset of $R(p,q)$ that has bottom right hand corner (c,d) with $c \geq p$ and $d = q$ and let

$$Z_j^{(y,z)}(p,q) = \sum_{m \in M(p,q)} p_{m,j}^{(y,z)}.$$

Then we can update $Q_j^{(y,z)}$ as follows

$$Q_j^{(y,z+1)} = R_j^{(y,z)} Q_j^{(y,z)} - R_j^{(y,z)} Z_j^{(y,z)}(y,z) + Z_j^{(y,z)}(y,z+w).$$

We now need to calculate $R_j^{(y,z)} Z_j^{(y,z)}(y,z)$ and $Z_j^{(y,z)}(y,z+w)$.

Let $R_j^{(y,z),w} = \prod_{s=z-w+1}^{z} R_j^{(y,s)}$. Then we can derive (c.f. the one dimensional case)

$$Z_j^{(y,z)}(y,z+w)) = (n-y)\left(\frac{R_j^{(y-1,z+w)} Z_j^{(y-1,z+w)}(y-1,z+w)}{(n-y+1)} + 1\right).$$

We also have

$$R_j^{(y,z)} Z_j^{(y,z)}(y,z)) = (n-y) R_j^{(y,z),w} \left(\frac{R_j^{(y-1,z)} Z_j^{(y-1,z)}(y-1,z)}{(n-y+1)} + 1\right).$$

Finally,

$$R_j^{(y,z),w} = \begin{cases} R_j^{(y,z)} & \text{if } z = 0 \\ R_j^{(y,z-1),w} R_j^{(y,z)} & \text{if } z < w \\ R_j^{(y,z-1),w} \frac{R_j^{(y,z)}}{R_j^{(y,z-w)}} & \text{if } w \leq z \leq n-1 \\ \frac{R_j^{(y,z-1),w}}{R_j^{(y,z-w)}} & \text{if } z > n-1 \end{cases}$$

The initial conditions needed are $Q_j^{(0,0)} = nw$ and

$$Q_j^{(y,0)} = \sum_{k=0}^{w-1} (n-y)\left(\frac{R_j^{(y-1,k)} Z_j^{(y-1,k)}(y-1,k)}{(n-y+1)} + 1\right).$$

With D distinct widths, we only have to run D distinct copies of the algorithm and sum the values of $Q_j^{(y,z)}$ for each copy. A minor complication arises at the boundaries of the z coordinate since multiple rectangles of different widths that go beyond the boundaries are in fact equivalent. One simple method of getting around this is to modify the algorithm for all values of widths w other than the largest width in such a way that $R_j^{(y,z)} Z_j^{(y,z)}(y,z) = 0$ for $z < w$, $Z_j^{(y,z)}(y, z + w) = 0$ for $z \geq n - w - 1$ and $Q_j^{(y,0)} = 0$.

If we allow all possible widths, the complexity of the algorithm is $O(Nn^3)$. The performance of the algorithm is then competitive against the class of k-rectangular tilings of the image for arbitrary k. Restricting the widths to be no more than W gives a complexity of $O(NWn^2)$ and an algorithm that is competitive against the class of k-rectangular tilings using rectangles with widths no more than W. Restricting the widths to powers of 2, gives a complexity of $O(Nn^2 \log n)$ and an algorithm that is competitive against the class of k-rectangular tilings using rectangles with widths of powers of 2.

4.3 Progressive Transmission

Performance of an extended model remains unchanged if progressive transmission is used, provided that the correct conditional probability distribution is used for the refinement levels. From Theorem 2, it follows that the bound of $k \log_2 \frac{m}{k}$, where k is the number of specialist models in U and m is the total number of specialist models, still holds when SBayes is used in a progressive transmission mode. The following example shows that even though the bound holds, the code length produced is sometimes different, unlike the case of the Bayesian probability allocation method.

Example 4 *Let* $a = (0.0, 0.0, 0.0, 1.0)$ *and* $b = (0.8, 0.0, 0.1, 0.1)$ *be two probability models over the alphabets* $(0, 1, 2, 3)$. *Consider a sequence of two symbols* $(x_0 = 3, x_1 = 3)$. *Let the class of specialists contain* $s_1 = (a, *)$, $s_2 = (b, b)$ *and* $s_3 = (*, a)$, *where* $*$ *means that the specialist is abstaining. The algorithm SBayes assigns a probability of 0.474 to the sequence. Now consider progressive transmission, where* $(x_{0,-1} = \{2, 3\}, x_{1,-1} = \{2, 3\})$ *is transmitted first followed by the refinement* $(x_0 = 3, x_1 = 3)$. *Using SBayes, the probability assigned is now 0.461.*

We now outline how to implement SBayes for the class of specialist rectangles for progressive transmission. First we consider the one dimensional case. The coarse resolution can be transmitted using the algorithm considered in the previous section. We will only describe one refinement level. The algorithms for further refinements are similar.

Let $R(i)$ be the class of all segments that contains the ith coefficient. Let

$$Q_j^i = \sum_{m \in R(i)} p_{m,j}^i$$

We have for each character u

$$a_{i,u} = \frac{\sum_{j=0}^{N-1} Q_j^i a_u^j / a_{u,-1}^j}{\sum_{j=0}^{N-1} Q_j^i}$$

where $a_{u,-1}^j$ is the probability of the lower resolution symbol. Let $M(q)$ be a subset of $R(q)$ containing segments that ends at q and let $N(q)$ be a subset of $R(q)$ containing segments that starts at q. Let

$$X_j^i(q) = \sum_{m \in M(q)} p_{m,j}^i$$

and

$$Y_j^i(q) = \sum_{m \in N(q)} p_{m,j}^i.$$

Let $R_j^i = a_{x_i}^j / (a_{x_i,-1}^j a_{i,x_i})$ and $R_{j,-1}^i = a_{x_i,-1}^j / (a_{i,x_i,-1})$ where $a_{x_i,-1}^j$ and $a_{i,x_i,-1}$ are the probabilities for the lower resolution symbol. Then we can update Q_j^i as follows

$$Q_j^{i+1} = R_j^i Q_j^i - R_j^i X_j^i(i) + Y_j^i(i+1)$$

We can obtain $R_j^{i+1} X_j^{i+1}(i+1)$ recursively by

$$R_j^{i+1} X_j^{i+1}(i+1) = R_j^{i+1} R_{j,-1}^{i+1} (R_j^i X_j^i(i) + 1).$$

The values for $Y_j^i(i+1)$ can be calculated recursively starting from $i = n-1$, where $Y_j^{n-1}(n) = R_{j,-1}^{n-1}$ and

$$Y_j^{i-1}(i) = R_{j,-1}^{i-1} (Y_j^i(i+1) + 1).$$

We now consider an $n \times n$ image and specialist models associated with rectangles of width w. Probability for the coarse resolution can be assigned as in the previous section.

Let $R(y,z)$ be the set of rectangles that contains the coordinate (y,z). Let

$$Q_j^{(y,z)} = \sum_{m \in R(y,z)} p_{m,j}^{(y,z)}$$

be the sum of the weights of the active specialist models associated with model j and the set $R(y,z)$ when the (y,z) coefficient is being processed. So we have, for each alphabet u

$$a_{(y,z),u} = \frac{\sum_{j=0}^{N-1} Q_j^{(y,z)} a_u^j / a_{u,-1}^j}{\sum_{j=0}^{N-1} Q_j^{(y,z)}}.$$

Let $R_j^{(y,z)} = a^j_{x_{(y,z)}}/(a^j_{x_{(y,z)},-1}a_{(y,z),x_{(y,z)}})$ and $R_{j,-1}^{(y,z)} = a^j_{x_{(y,z)},-1}/(a_{(y,z),x_{(y,z)},-1})$ where $a^j_{x_{(y,z)},-1}$ and $a_{(y,z),x_{(y,z)},-1}$ are the probabilities for the lower resolution symbol. Let $M(p,q)$ be a subset of $R(p,q)$ that has bottom right hand corner (c,d) with $c \geq p$ and $d = q$ and let

$$Z_j^{(y,z)}(p,q) = \sum_{m \in M(p,q)} p_{m,j}^{(y,z)}.$$

As in the previous section, we can update $Q_j^{(y,z)}$ as follows

$$Q_j^{(y,z+1)} = R_j^{(y,z)} Q_j^{(y,z)} - R_j^{(y,z)} Z_j^{(y,z)}(y,z) + Z_j^{(y,z)}(y,z+w).$$

We now need to calculate $R_j^{(y,z)} Z_j^{(y,z)}(y,z))$ and $Z_j^{(y,z)}(y,z+w)$.

The values $R_j^{(y,z)} Z_j^{(y,z)}(y,z)$ and $Z_j^{(y,z)}(y,z+w)$ can be updated in a manner similar to the one dimensional case. We have

$$Z_j^{(y+1,z)}(y+1,z+w) = R_j^{(y,z+w),w} Z_j^{(y,z+w)} - R_j^{(u,z+w),w} X_j^{(y,z+w)}(y,z+w) + Y_j^{(y,z+w)}(y+1,z+w)$$

where

$$R^{(y+1,z+w),w} X_j^{(y+1,z+w)}(y+1,z+w) = R_j^{(y+1,z+w),w} R_{j,-1}^{(y+1,z+w),w}(R_j^{(y,z+w),w} X_j^{(y,z+w)}(y,z+w)+1)$$

and

$$Y_j^{(y-1,z+w)}(y,z+w) = R_{j,-1}^{(y-1,z+w),w}(Y_j^{(y,z+w)}(y+1,z+w)+1).$$

Similarly,

$$R_j^{(y+1,z)} Z_j^{(y+1,z)}(y+1,z) = R_j^{(y+1,z),w}(R_j^{(y,z),w} Z_j^{(y,z)} - R_j^{(u,z),w} X_j^{(y,z)}(y,z) + Y_j^{(y,z)}(y+1,z))$$

where

$$R^{(y+1,z),w} X_j^{(y+1,z)}(y+1,z) = R_j^{(y+1,z),w} R_{j,-1}^{(y+1,z),w}(R_j^{(y,z),w} X_j^{(y,z)}(y,z)+1)$$

and

$$Y_j^{(y-1,z)}(y,z) = R_{j,-1}^{(y-1,z),w}(Y_j^{(y,z)}(y+1,z)+1).$$

The complexity of the algorithm for progressive transmission is $O(LDNn^2)$ where L is the number of resolutions and D is the number of discrete widths considered.

5 Discussion

In this paper, we have concentrated on compression algorithms that use a single scalar quantizer for quantizing all of the transform coefficients of an image. By utilizing a more sophisticated quantizer such as a trellis-coded quantizer [8], improved compression may be possible. This is readily done with forward adaptation methods, for example with quadtrees and optimal pruning. We do not know how to obtain *efficient* forward adaptation methods with good bounds on the redundancy for the class of k-rectangular tilings of an image. Such forward adaptation methods may allow the use of sophisticated quantization methods in conjunction with this class of models.

6 Conclusions

We have introduced a class of probability models for image, the k-rectangular tilings of an image, as a comparison class to be used for image compression. We give a sequential probability assignment algorithm with a redundancy of $O(k \log \frac{Nn}{k})$ relative to the class of k-rectangular tilings of an $n \times n$ image using N probability models for the coefficients. The computational complexity of the algorithm is $O(Nn^3)$. With widths of rectangles restricted to one of D values, the computational complexity of the algorithm reduces to $O(DNn^2)$. This gives an algorithm with a computational complexity of $O(WNn^2)$ that is competitive against the class of k-rectangular tilings using rectangles of width W or less. If the widths are of powers of 2, the computational complexity of the algorithm is $O(Nn^2 \log n)$, which is comparable to the complexity of sequential probability assignment using a quadtree. We also show that progressive transmission is also possible with the same bound on the redundancy and similar computational complexity.

7 Acknowledgment

The author would like to thank Phil Long for helpful discussions. The author would also like to thank the anonymous reviewers whose comments helped to improve the presentation of the paper.

References

[1] Avrim Blum. Empirical support for winnow and weighted-majority based algorithms: results on a

calendar scheduling domain. *Machine Learning*, 26:5–23, 1997.

[2] C. Chrysafis and A. Ortega. Efficient context-based lossy wavelet image coding. In *Proc. of Data Compression Conference*, Snowbird, Utah, 1997.

[3] W. Equitz and T. Cover. Successive refinement of information. *IEEE Transactions on Information Theory*, 37(2):269–275, March 1991.

[4] Yoav Freund, Robert E. Schapire, Yoram Singer, and Manfred Warmuth. Using and combining predictors that specialize. In *Proceedings of the Twenty-Ninth Annual ACM Symposium on the Theory of Computing*, El Paso, Texas, 1997.

[5] Robert M. Gray. *Source Coding Theory*. Kluwer Academic Publishers, 1990.

[6] David P. Helmbold and Robert E. Schapire. Predicting nearly as well as the best pruning of a decision tree. *Machine Learning*, 27(1):51–68, 1997.

[7] M. Herbster and M. Warmuth. Tracking the best expert. *Machine Learning*, 32(2), August 1998.

[8] R. L. Joshi, H. Jafarkhani, J. H. Kasner, T. R. Fischer, N. Farvardin, M. W. Marcellin, and R. H. Bamberger. Comparison of different methods of classification in subband coding of images. *IEEE Transactions on Image Processing*, 6:1473–1486, November 1997.

[9] Wee Sun Lee. Trees, windows and tiles for wavelet image compression. In *Data Compression Conference*, Snowbird, Utah, 2000.

[10] S. M. LoPresto, K. Ramchandran, and M. T. Orchard. Image coding based on mixture modelling of wavelet coefficients and a fast estimation-quantization framework. In *Proceedings of the Data Compression Conference*, pages 221–230, Snowbird, Utah, 1997.

[11] N. Merhav. On the minimum description length principle for sources with piecewise constant parameters. *IEEE Transactions on Information Theory*, 39(6):1962–1967, November 1993.

[12] N. Merhav and M. Feder. Universal prediction. *IEEE Transactions on Information Theory*, 44(6):2124–2147, October 1999.

[13] ISO/IEC JTC/SC29/WG1 N1385. JPEG2000 requirements and profiles version 6.0, July 1999.

[14] A. Said and W.A. Pearlman. A new, fast, and efficient image codec based on set partitioning in hierarchical trees. *IEEE Trans. on Circuit and Systems for Video Technology*, 6(3):243–249, 1996.

[15] Jerome M. Shapiro. Embedded image coding using zerotrees of wavelet coefficients. *IEEE Transactions on Signal Processing*, 41(12):3445–3462, December 1993.

[16] Paul A. J. Volf and Frans M. J. Willems. Switching between two universal source coding algorithms. In *Data Compression Conference*, pages 491–500, Snowbird Utah, 1998.

[17] V. Vovk. Derandomizing stochastic prediction strategies. In *Proceedings of the Tenth Annual Conference on Computational Learning Theory*, pages 32–43, Nashville, Tennessee, 1997.

[18] F. M. J. Willems, Y. M. Shtarkov, and T. J. Tjalken. The context-tree weighting method: Basic properties. *IEEE Trans. on Information Theory*, 41(3):653–664, May 1995.

[19] Frans M. J. Willems. Coding for a binary independent piecewise-identically-distributed source. *IEEE Transactions on Information Theory*, 42(6):2210–2217, November 1996.

[20] Xiaolin Wu. Context quantization with Fisher discriminant for adaptive embedded wavelet image coding. In *Proceedings Data Compression Conference*, pages 102–111, Snowbird, Utah, 1999.