Context-Inclusive Approach to
Speed-up Function Evaluation for Statistical Queries

A project
submitted to the faculty of the graduate school
at the University of Minnesota

By
Vijay M Gandhi

In partial fulfillment of the requirements
for the degree of
Master of Science

September 2007

Advisor: Dr. Shashi Shekhar
Department of Computer Science and Engineering
Acknowledgements

I sincerely thank my advisor Dr. Shashi Shekhar for guiding and providing ideas throughout this project. I take this opportunity to thank Dr. Shashi Shekhar for letting me work with his group and guide me throughout my graduate studies. I thank Dr. Bradley Carlin and Dr. Jaideep Srivastava for agreeing to participate in my project defense.

I thank James Kang, and Mete Celik for collaborating with me on the research papers that were published out of this project. I thank Spatial Database group at the University of Minnesota for their help in my research work and providing feedback to improve my presentation. I also thank Dr. Junchang Ju, Dr. Sucharita Gopal, and Dr. Eric Kolaczyk at the Boston University for introducing me to the application domain. A special thanks to Kim Koffolt for improving the readability of my research papers.

Most of all, I dedicate my graduate degree to my mother, Neeta Gandhi, for her selfless love and support throughout my life. Without her support, I couldn’t have come so far. I am proud to have my brother, Jay Gandhi, and friend Naveen Nagarajan, whom I can always rely on for help and encouragement.
Abstract

Many statistical queries such as maximum likelihood estimation involve finding the best candidate model given a set of candidate models and a quality estimation function. This problem is common in important applications like land-use classification at multiple spatial resolutions from remote sensing raster data. Such a problem is computationally challenging due to the significant computation cost to evaluate the quality estimation function for each candidate model. A recently proposed method of multiscale, multigranular classification has high computational overhead of function evaluation for various candidate models independently before comparison. In contrast, we propose a context-inclusive approach that controls the computational overhead based on the context, i.e. the value of the quality estimation function for the best candidate model so far. Experimental results using land-use classification at multiple spatial resolutions from satellite imagery show that the proposed approach reduces the computational cost significantly while providing comparable classification accuracy.
Contents

1 Project Overview and Scope 5

2 Introduction 5

3 Background 8
  3.1 Expectation Maximization .......................... 8
  3.2 Execution Trace .................................... 8

4 Approaches 10
  4.1 Context-Exclusive Approach .......................... 11
  4.2 Context-Inclusive Approach .......................... 11
    4.2.1 Theoretical Analysis ................................. 13
  4.3 Limiting Factor ..................................... 14

5 Experiments 15

6 Discussion and Future Work 18

References 18

Appendix A: Expectation Maximization Example 20

Appendix B: Parallel Implementation 21
1 Project Overview and Scope

As a Research Assistant to Dr. Shashi Shekhar in the department of Computer Science and Engineering, I worked on a project related to Multiscale Multigranular Image Classification. My role was to explore ways to improve the performance of the algorithm. Table 1 lists various techniques I explored. The scope of my Plan B project is limited to the Context-Inclusive approach to improve the computational efficiency. This report also provides details about the black box tuning. Details regarding the parallel implementation are provided as an appendix. Code conversion from MATLAB to C is not discussed. Please also note that most of the text in this report has been taken from the papers published as a part of the research carried out with this project.

<table>
<thead>
<tr>
<th>Approach</th>
<th>Description</th>
<th>Result</th>
<th>Effort (in hours)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Black box tuning</td>
<td>Change in precision of \textit{limiting factor}</td>
<td>Reduced computation time by 50% with accuracy of 98%</td>
<td>20</td>
</tr>
<tr>
<td>Code Conversion</td>
<td>Entire code was converted from MATLAB to C</td>
<td>Reduced computation time by 50%</td>
<td>120</td>
</tr>
<tr>
<td>Parallelization</td>
<td>Parallelized the code using Cray X1 and UPC programming language</td>
<td>Obtained linear speedup of 6.65 using 8 processors</td>
<td>40</td>
</tr>
<tr>
<td>Context-Inclusive approach</td>
<td>Change in algorithm</td>
<td>Reduced computation time by 53% with 98% accuracy</td>
<td>100</td>
</tr>
</tbody>
</table>

Table 1: Approaches to improve computational efficiency of Multiscale Multigranular Image Classification

2 Introduction

We are interested in a probabilistic statistical query to find the preeminent candidate model from a set of candidate models using a quality estimation function. In my project, such a problem is referred as the \textit{best candidate model problem}. Formally, it can be stated as follows: Given a set of candidate models and a function to evaluate the quality of each candidate model, the goal is to find the best candidate model probabilistically. The evaluation of this measure is generally very expensive and thus minimizing the computation time is a key objective.

One important example of the \textit{best candidate model problem} is in classification of a spectral image, obtained from

\footnotesize
\begin{itemize}
  \item Parallelizing Multi-scale and Multi-granular Spatial Data Mining Algorithm, Vijay Gandhi, Mete Celik, Shashi Shekhar. Partitioned Global Address Space Programming Models Conference, 2006
\end{itemize}
a satellite, with domain-specific labels to produce a thematic map. Thematic maps are widely used in applications including agricultural monitoring, land cover change analysis, and environmental assessment. Examples in land cover change can be seen in Figure 1, where 217 square miles of Louisiana’s coastal lands were transformed to water after hurricanes Katrina and Rita (Figure 1a); the deforestation in Brazil causing the loss of 150,000 sq. km. of forest between May 2000 and August 2006 (Figure 1b); and urban sprawl in Atlanta, GA between 1976 and 1992 (Figure 1c).

![Figure 1: Land Cover Change Examples. (Best viewed in color)](image)

Image classification at multiple spatial resolutions is an important application of spatial data mining. For example, NASA’s Earth observation systems obtain a spectral image of land-use, which is then classified at multiple resolutions. The best candidate model problem to find the best classification label can be considered as a parameter estimation problem. Since estimating parameters at a spatial region is an important function in spatial data mining, the best candidate model problem is a sub-class of spatial data mining.

Figures 2 and 3 gives an example of a multiscale, multigranular image classification based on land usage. Figure 2 gives the input that includes a set of domain-specific labels (also called classes) logically grouped as a hierarchy (Figure 2a), and the values of each specific class: conifer, hardwood, brush, and grass (Figures 2c-f). The values of specific class are derived from a synthetic remotely-sensed satellite image [8], (Figure 2b) that has been provided in this report for completeness. The goal is to classify each pixel in the satellite image to one of the labels based on a quality measure called likelihood. The likelihood measure is calculated using a function called Expectation Maximization (EM) [2] which is expensive because of the large number of iterations required until convergence. Also, multiple scales are defined implicitly in powers of 2 (i.e. $2 \times 2, 4 \times 4, \ldots, 2^{n-1} \times 2^{n-1}$, where $n$ is the size of the image). An example output in shown in Figure 3 having a set of classified images at scales $1 \times 1, 2 \times 2, 4 \times 4,$ and $64 \times 64$.

Calculating the likelihood of each candidate model makes the problem computationally expensive. For instance,
the work proposed by [8], takes about 7 hours of computation time to classify an image of size 512 x 512 pixels with 12 labels at varying spatial resolutions. About 80% of the total computation time is consumed to find the quality measure for each candidate model. Thus, as the image size grows the computation time increases, which makes this problem challenging.

Numerous studies in remote sensing have been done for multi-resolution land-use classification (e.g., [6, 9, 11]). See [15] for a detailed discussion on various methods for multi-resolution classification. A statistical method to classify an image at varying spatial and categorical resolutions was proposed in [8]. This approach is context-exclusive, based on using a query tree to identify each candidate model independently. The maximum likelihood is used as a set operator among quality measures for each candidate model, thus making it necessary to analyze all candidate models together. The result is very high computation costs to identify the preeminent candidate model.

Real and synthetic raster data sets from remote sensing were used for the experimental studies. Contributions can be summarized as follows:

1. A context-inclusive function evaluation approach that exploits the natural relationship among all candidate models to distinguish the preeminent candidate model.
2. An insight to further reduce computation costs using a limiting factor to iteratively monitor the quality measures of all candidate models and exit when the appropriate candidate model is discovered.
3. Experimental evaluation to compare the approach with a previous approach [8].

The rest of the report is organized as follows: A background on the EM algorithm and its application to the problem is described in Section 3. Section 4 gives a detailed overview of the approach along with the major differences with previous work. As part of proposed approaches, theoretical analysis of the context inclusive approach with an upper bound is described in Section 4.2.1. Experimental results to compare the previous and the proposed approach are given in Section 5. Finally, Section 6 concludes this report with a summary and future work.

3 Background

This section presents a general overview of Expectation Maximization and its application to the best candidate problem. Also, an execution trace example is shown.

3.1 Expectation Maximization

Previous analysis of multiscale, multigranular classification focused on the mixture discriminant analysis (MDA) model for estimating class regions from satellite imagery [7]. Each class is modeled with a simple density component of an overall mixture density using finite mixture models, which are used quite often in the remote sensing and image analysis community. This framework is i.e., each land cover class as a mixture of subclasses of multivariate Gaussian distributions, is used in this land cover classification algorithm. Expectation Maximization algorithm is used to estimate the proportions of corresponding specific classes for a non-specific class. These proportions are in turn used to calculate the likelihood of the non-specific class.

3.2 Execution Trace

There are many implementations of the Expectation Maximization algorithm [2]. Algorithm 1 presents the
Algorithm 1: Expectation Maximization

1: Function EM(class Non-Specific, spatial, region quad)

2: repeat
3:   Step 1 Initialize the proportion of each corresponding specific class in the spatial region
4:   Step 2 Multiply likelihood at each pixel in the quad by corresponding specific class proportion
5:   Step 3 Add the likelihood at corresponding pixel
6:   Step 4 Divide the value in Step 2 by value in Step 3 at corresponding pixel
7:   Step 5 Average the likelihood in the quad for each specific class and consider these to be new proportions
8: until the required accuracy is achieved
9: return sum of the product of proportions and likelihood Maximum Likelihood

<table>
<thead>
<tr>
<th>Specific Class</th>
<th>Pixel 1</th>
<th>Pixel 2</th>
<th>Pixel 3</th>
<th>Pixel 4</th>
</tr>
</thead>
<tbody>
<tr>
<td>conifer (X1)</td>
<td>0.0077</td>
<td>0.0509</td>
<td>0.0067</td>
<td>0.0034</td>
</tr>
<tr>
<td>hardwood (X2)</td>
<td>0.0031</td>
<td>0.0971</td>
<td>0.2170</td>
<td>0.0141</td>
</tr>
</tbody>
</table>

Table 2: Likelihood values (in $10^{-4}$) for specific classes of Forest

Expectation Maximization algorithm that is used in the problem domain as described in [8]. we present the algorithm to discuss the computation aspect but not the correctness. The input to Algorithm 1 is the Non-Specific class (e.g. forest) to determine the maximum likelihood of its specific class (e.g. conifer and hardwood) proportions and the spatial region (e.g. a quad of size 2 x 2 pixels). There are five steps within this algorithm: (1) the proportion size of each specific class under the given non-specific class initialized where all proportions are equal to one (Line 3 of Algorithm 1), (2) the likelihood of each pixel in the spatial region is multiplied to each class proportion (Line 4 of Algorithm 1), (3) the likelihood values for each non-specific class are added together (Line 5 of Algorithm 1), (4) the multiplied values in step 2 are divided by the summed up values in step 3 (Line 6 of Algorithm 1), and (5) the average is calculated for each proportion or specific class (Line 7 of Algorithm 1). This process is repeated until the desired accuracy of the proportions is acquired. Accuracy is determined by the amount of change in the likelihood value based on the proportion size between iterations.

To give an example, consider the use of the Expectation Maximization algorithm for a non-specific class forest, calculated for a quad of size 2 x 2 pixels. The initial likelihood values of specific classes corresponding to forest

| EM Iteration | Input Proportions (P) | Output Proportions (Q) | Error ($||P-Q||/||P||$) |
|--------------|----------------------|-----------------------|------------------------|
| 1            | [0.5000,0.5000]      | [0.3202,0.6798]       | 0.2008                 |
| 2            | [0.3202,0.6798]      | [0.2135,0.7865]       | 0.1111                 |
| 3            | [0.2135,0.7865]      | [0.1494,0.8506]       | 0.0666                 |
| 4            | [0.1494,0.8506]      | [0.1088,0.8912]       | 0.0094                 |

Table 3: Execution Trace of Algorithm 1
are represented by vectors in Table 2. Assuming equal initial proportions [0.5,0.5] of conifer and hardwood, the Expectation Maximization algorithm finds the best values for these proportions iteratively. Once the proportions of corresponding specific classes are found, the likelihood value for the non-specific class is given by the sum of the product of the proportions and the initial likelihood values. In practice, the logarithm value of the likelihood is taken for numerical stability. In the example, the EM algorithm continues for 4 iterations, until the desired accuracy is reached, i.e., error is less than 0.01. The execution trace of Algorithm 1 specific to this example is provided in Table 3. A detailed step by step execution trace is provided in the Appendix.

4 Approaches

In this subsection, we present the proposed approach to address the best candidate model problem which may be represented by two tree based methods. The first tree based method uses information from the ancestors and siblings of a query tree to identify the appropriate strategy. Figure 4a gives an example of a query tree where the root node is a set/relational operator, the interior node is a table transformation, and the leaf is a single table. An example of a set operator at the root of this query tree is MAX. A variant to the query tree is an instance-level syntax tree where the main distinction is that the former has the whole table or relation as the leaf and the latter has multiple leaves consisting of individual tuples in a table (Figure 4b). Similar to the query tree, the root can also be represented as a MAX set operation. However, the instance-level syntax tree has many more children than the query tree where each node is represented by a quality measure (i.e., likelihood function) and each respective child as a distinct candidate model.

Section 4.1 reviews a previous context exclusive approach using a query tree. section 3.2 explains the proposed approach to reduce the computational complexity of the best candidate model problem using an instance-level syntax tree as it is applied to a land-use classification problem in the remote sensing domain. In section 4.2, we discuss the insight to further reduce the computation time of identifying the best candidate model.
4.1 Context-Exclusive Approach

Algorithm 2 Context-Exclusive Approach

1: Function contextExclusive(set Cand)
2: Set of Candidate and Likelihood Values $SCL \leftarrow \{(\emptyset, \emptyset)\}$
3: for each candidate $c \in Cand$ do
4: \quad $SCL \leftarrow SCL \cup (c, EM(c))$
5: end for
6: return $\arg \max(SCL), \text{MAX}(SCL)$

A query tree may evaluate an entire table independent from other tables and this relationship is referred as context-exclusive. Algorithm 2 presents the pseudo code of the context-exclusive approach from [8]. The input to Algorithm 2 is the set of non-specific class candidates and the output is the arg max candidate in $Cand$ and its corresponding likelihood value (i.e. maximum value in $SCL$). The main objective in Algorithm 2 is to obtain the candidate classification for a spatial region. Each candidate model in $Cand$ is analyzed independently to obtain its likelihood for a spatial region by executing EM from Algorithm 1 (Line 4 in Algorithm 2). The candidate model containing the maximum or largest likelihood of all candidates in $Cand$ is declared the best candidate model for a spatial region (Line 7 in Algorithm 2).

4.2 Context-Inclusive Approach

Proposed approach utilizes an instance-level syntax tree where each tuple is evaluated together to obtain the optimal candidate model, a relationship which we refer to as context-inclusive. Figure 4c gives an example of the instance-level syntax tree as it is applied within the remote sensing domain. The root node of the instance-level syntax tree represents the maximum likelihood set operator and the interior nodes are the likelihood function quality measures for each respective child consisting of candidate model tuples. Using remote sensing terminology, each candidate model tuple represents a classification consisting of either a specific (i.e., conifer or hardwood) or a general (i.e., forest) class (see Figure 2a). The quality measure for a specific class is a single value whereas a general class consists of several proportions of specific classes. For example, a general class of type forest may have several proportions of conifer and hardwood trees. A computationally very expensive function is used (i.e., EM) to identify the likelihood value for each candidate model. The main objective is to find the maximum likelihood value or best candidate model to represent a land-use classification.

Algorithm 3 presents the pseudo code for the context-inclusive approach to find the optimal candidate model as it is applied to the domain of remote sensing. The input to Algorithm 3 is the set of non-specific candidate models $Cand$ for a spatial region, and the output is the maximum likelihood candidate models in $BestCand$, and the best likelihood value in $ML_c, and$. Initially, a candidate is chosen among the set of non-specific models and its
Algorithm 3 Context-Inclusive Approach

1: Function CONTEXTINCLUSIVE(set Cand)
2: Randomly select $c_{\text{cand}}$ from Cand
3: $\text{ML}_{\text{cand}} \leftarrow \text{EM}(c_{\text{cand}})$
4: $\text{BestCand} \leftarrow c_{\text{cand}}$
5: for each $c \in \text{Cand}$ AND $c \neq c_{\text{cand}}$ do
6:  repeat
7:    Step 1 Initialize the proportion of each corresponding specific class under the given non-specific class
8:    Step 2 Multiply each likelihood by corresponding specific class proportion
9:    Step 3 Add the likelihood at corresponding pixel
10:   Step 4 Divide the value in step 2 by corresponding the value in Step 3
11:   Step 5 Average the likelihood for each specific class
12:   if $\text{likelihood} > \text{ML}_{\text{cand}}$ then
13:      $\text{BestCand} \leftarrow c$
14:      $\text{ML}_{\text{cand}} \leftarrow \text{likelihood}$
15:   end if
16: until the required accuracy is achieved OR new $\text{BestCand}$ is found
17: end for
18: return $\text{BestCand}, \text{ML}_{\text{cand}}$

maximum likelihood is found by executing EM from Algorithm 1 (Lines 2-3 in Algorithm 3). The best candidate for the spatial region is initially set as the randomly chosen candidate (Line 4 in Algorithm 3). Then, for each non-specific candidate, except the randomly chosen candidate, the five steps of the EM algorithm are executed (Lines 5-11 in Algorithm 3). Unlike EM in algorithm 1, if the likelihood value is larger than the initial randomly chosen candidate, then a new best candidate is found and exits the EM portion in this algorithm (Lines 12-16 in Algorithm 3). The evaluation of EM for a class based on the best likelihood value so far is the main distinction between the Context Inclusive and Context Exclusive Approaches. Experimental analysis (section 5) has shown that the removal of these iterations maintains high accuracy in this classification method. After each candidate has been evaluated, the best candidate having the largest likelihood value is returned (Line 18 in Algorithm 3).

To illustrate with an example, consider the hierarchy defined in Figure 2a. Computing the likelihood value for a specific class (conifer, hardwood, brush, grass) is not expensive while computing the likelihood value for a general class (forest, non-forest, vegetation) is. For a spatial region, initially the likelihood of specific classes are determined to find the one with the highest value. Assuming the class with the highest value was brush, the candidate models to be compared are: brush, forest, non-forest, vegetation. Assume the maximum likelihood values for brush, forest, non-forest, vegetation were supposed to be -53.5828, -53.3381, -51.3985, and -56.2506 respectively. As shown in Figure 5, EM runs iteratively until the maximum value for each of the non-specific class is reached. In this case, the total number of EM iterations taken are 83 (46 for vegetation, 34 for forest, 3 for
non-forest). Once the maximum likelihoods are calculated for each of the non-specific classes they are compared to find the best. In the example here, the maximum likelihood value of non-forest is selected. In the context-inclusive approach, illustrated in Figure 6, the likelihood value for a non-specific class is calculated until it exceeds the current best likelihood value. To start with, the likelihood value of brush is the best and the likelihood of others are yet to be determined. The likelihood for vegetation is evaluated next using the EM. At each iterative step in EM, the current likelihood value of vegetation is compared to the likelihood value of the best so far i.e., brush. As soon as the likelihood of vegetation exceeds that of brush, EM stops iterating further. In the example, the likelihood of vegetation never exceeds that of brush. So, EM iterates until the desired accuracy is reached. However, when the likelihood of forest is calculated next, EM iterates only 4 steps, to the point when the likelihood of forest, -53.4090, is just greater than the likelihood of the current best, that is, of brush, -53.5828. Compared to the context-exclusive approach, EM for forest saved 30 steps. In the next step, EM is used to calculate the likelihood of non-forest class, -51.3985, that takes only 1 iteration because it exceeds the value of the current best, that is, of forest, -53.4090. Compared to the context-exclusive approach, EM for non-forest takes 2 fewer steps. Overall, in this example, there was a saving of 32 EM iterations. This difference is quite significant when the size of the spatial region is high.

4.2.1 Theoretical Analysis

Algorithm 4.2.1 presents the Context Inclusive approach if a theoretical upper bound exists. A theoretical upper bound is calculated for each non-specific candidate to ensure that the correct candidates are pruned from the candidate set (Theorem 1). By determining the upper bound on the likelihood value of a non-specific class (Line 7 of Algorithm 4.2.1), it is possible to prune candidates whose upper bound is lower than the current best likelihood value. This ensures that those candidates that have an upper bound less than the current best candidate can not be the best candidate and are then pruned (Lines 8-9 of Algorithm 4.2.1). For all other candidates, their respective maximum likelihoods are found through the EM algorithm and compared against the current best
Algorithm 4 Context-Inclusive Approach with Theoretical Upper Bound

1: Function contextInclusiveThUpper(set Cand)
2: Randomly select c_{cand} from Cand
3: ML_{cand} ← EM(c_{cand})
4: BestCand ← c_{cand}
5: for each c ∈ Cand AND c ≠ c_{cand} do
6:  ThUpper ← Theoretical Upper Bound for Cand c
7:  if ThUpper < ML_{cand} then
8:    c is pruned from Cand
9:  else
10:   c_{ML} ← EM(c)
11:  if c_{ML} > ML_{cand} then
12:    BestCand ← c
13:   ML_{cand} ← c_{ML}
14: end if
15: end if
16: end for
17: return BestCand, ML_{cand}

candidate (Lines 12-15 of Algorithm 4.2.1). Finally, the best candidate having the largest or maximum likelihood of all other candidates is returned (Line 18 of Algorithm 4.2.1).

**Theorem 1** Context Inclusive is correct such that each pixel in the image is classified with the maximum likelihood or best candidate class from the user-defined concept hierarchy.

**Proof** Based on Algorithm 4.2.1, a theoretical upper bound is assigned to each non-specific class candidate except for the randomly chosen candidate because the convergence likelihood is found in Lines 2-6. The randomly chosen candidate is assigned as the initial best candidate. If the upper bound for every other candidate is less than the current best candidate’s likelihood, then the candidate will not ever have a higher likelihood value and can then be pruned from the list (Lines 5-8 of Algorithm 4.2.1). All other candidates having a higher upper bound than the current best candidate may have a higher actual likelihood value (from EM), which is then assigned as the best candidate (Lines 10-14 of Algorithm 4.2.1). Thus, the best correct candidate will be retrieved from the candidate list. □

4.3 Limiting Factor

we discovered an insight to reduce the computation time for the exit criteria (limiting factor) in the EM used in the previous approach [8]. The objective in this EM is to determine an accurate representation of the proportion sizes of each specific component in a general class. This EM computes the quality measure at each iteration to be
used as input for subsequent iterations. For the quality measure at each iteration, a limiting factor is introduced to determine if the current measure represents the best proportions for the general class. In [8], the limiting factor was used at a very fine level of detail. Since the proportions are based on the underlying distributions of specific classes, we varied this limiting factor at lower levels, which reduced the number of iterations significantly while maintaining a consistent level of accuracy. Our experiments support the claim that using a lower limiting factor will reduce the computations without sacrificing a high level of accuracy.

5 Experiments

The primary goal of the experiments was to compare the context-inclusive approach with the context-exclusive approach in terms of computation and accuracy. A secondary goal was to study the effect of change in limiting factor on computation and accuracy. Figure 5 provides a schematic representation of the experimental design. All experiments were performed on two different datasets: (1) A synthetic input image of size 128 x 128 pixels, 7 total classes, and 3 general classes (Figure 2); and (2) A real dataset of Plymouth County, Massachusetts, consisting of an input image of size 128 x 128 pixels, 12 total classes, and 4 general classes. Outputs were obtained for varying spatial scales (resolutions). Spatial scale of 1 corresponds to spatial regions of size 2 x 2 pixels each and increases by a power of 2 for each subsequent spatial resolution i.e., a spatial scale of 6 indicates that the regions are of size 64 x 64 pixels each. All experiments were performed on an UltraSparc IIwe 1.1 GHz processor with 1GB of RAM.

As discussed in section 1, evaluation of the quality measure takes up most of the time. Hence the number of iterations in EM may be used to evaluate the computation of the context-inclusive and context-exclusive algorithms. Experimental results of both context-inclusive and context-exclusive approaches at multiple spatial scales on Dataset 1 and Dataset 2 are shown in Figures 8 and 10, respectively. Increase in spatial scale increases the number of iterations because of the increase in size of regions. Note that Dataset 2 has more general classes than Dataset 1, and thus is more computationally expensive. Experimental results show that the number of iterations is less for context-inclusive approach because the EM algorithm exits as soon as a candidate model when the best
Figure 8: Iterations in EM for Dataset 1

Figure 9: CPU Time for Dataset 1

Figure 10: Iterations in EM for Dataset 2

Figure 11: CPU Time for Dataset 2
likelihood is found. Compared to the context-exclusive approach, at the spatial scale of 6, the number of iterations with the context-inclusive approach reduces by 67.6% for Dataset 1 and by 61.45% for Dataset 2.

Figures 9 and 11 provide the execution time taken for Datasets 1 and 2, respectively. Since the number of iterations taken by the context-inclusive approach is less than that for the context-exclusive approach, the execution time for the context-inclusive approach is lesser. At the spatial scale of 6, execution time for the context-inclusive approach, as compared to the context-exclusive approach, is reduced by 53.34% and 47.48% for Dataset 1 and Dataset 2 respectively. The speedup is obtained without sacrificing accuracy. Table 4 provides the relative accuracy of the context-inclusive approach as compared to the results from the context-exclusive approach. For both the datasets, relative accuracy is always more than 98%.

EM computes the mixture proportions (likelihood) iteratively. The limiting factor for EM can be varied based on the desired accuracy for the final mixture proportion. Figures 12 and 13 compare the number of iterations and execution time, respectively, for different limiting factors. The total number of iterations in EM and execution time decreases linearly with the decrease in limiting factor. As shown in Figure 13 for Dataset 2, the execution time decreases from 133 minutes to 55 minutes for a change in limiting factor from 0.00001 to 0.01. Table 5 provides the accuracy results for a limiting factor of 0.01 as compared to the limiting factor of 0.00001 at different scales. These results show that the cost-effective approach does not sacrifice accuracy.

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Spatial Scale</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>1</td>
</tr>
<tr>
<td>Dataset 2</td>
<td>99.59</td>
</tr>
</tbody>
</table>

Table 4: Relative Accuracy for Context-Inclusive Approach
In this report, we presented a context-inclusive approach in an instance-level syntax tree where each tuple is evaluated together to obtain the optimal candidate model. This approach differs from previous work which uses a context-exclusive method in a query tree [8]. Although a context-inclusive approach has been applied to query trees before [12], it was never applied to an instance-level syntax trees. A limiting factor is also introduced that reduces the number of iterations toward convergence in EM. The experiments show that the approach reduces the computational complexity over the previous approach while maintaining comparable classification accuracy.

Other types of context may be explored. For example, spatial context i.e., the correlation of a variable with space [13], may be used. Classification of spatial data based on an extended regression model called the Spatial Auto-regression model (SAR) is provided in [14].

As discussed, most of the execution time is spent in calculating the quality measure using EM. EM is used to find the best candidate model; more specifically, EM is used to find the best Gaussian mixture model in the case of land-use classification. We plan to explore faster implementations of EM such as [4, 5, 10].

Studies in upper bound analysis are often limited in obtaining a tight bound on the maximum likelihood (e.g. [1]). In the preliminary work, we tried applying [1] to calculate an upper bound in the example. But, the calculation of the upper bound turned out to be expensive and did not provide adequate pruning. The availability of cheap upper bound calculation methods with efficient pruning could speed-up the algorithm. We plan to explore this challenging problem to obtain an efficient context inclusive approach with 100% accuracy and thus improve the current accuracy level of 99% accuracy. Previously, we explored different ways of parallelizing the context-exclusive approach [3]. We plan to work on applying similar techniques to the context-inclusive approach.

Finally, the proposed approach uses a bottom-up strategy where information from finer spatial scales is used for coarser spatial scales. Enhancements may be made to consider a top-down approach where a pruning procedure may be used to reduce computation.

References


Appendix A: Expectation Maximization Example

This section demonstrates the Expectation Maximization algorithm used in the report with an example. Input to the EM algorithm is the non-specific class, forest, and the likelihood values of corresponding specific classes, conifer and hardwood, for each pixel in the spatial region. Table 6 provides the likelihood values of conifer and hardwood for a spatial region of 2 x 2 pixels. The initial proportions are assumed to be (0.5,0.5). Steps 2 through 5 of algorithm 1 are executed until the error value is less than desired limit of 0.01. The result from each step is provided in Table 7. The final proportions (0.1088,0.8912) are used to calculate the likelihood of the input non-specific class. Likelihood is calculated as the sum of product of the proportions and the corresponding likelihood values of specific classes in the spatial region. Logarithm value of the likelihood is taken for numerical stability.

<table>
<thead>
<tr>
<th>Specific Class</th>
<th>Pixel 1</th>
<th>Pixel 2</th>
<th>Pixel 3</th>
<th>Pixel 4</th>
</tr>
</thead>
<tbody>
<tr>
<td>conifer (X1)</td>
<td>0.0077</td>
<td>0.0509</td>
<td>0.0067</td>
<td>0.0034</td>
</tr>
<tr>
<td>hardwood (X2)</td>
<td>0.0031</td>
<td>0.0971</td>
<td>0.2170</td>
<td>0.0141</td>
</tr>
</tbody>
</table>

Table 6: Likelihood values (in $10^{-4}$) for specific classes of forest
<table>
<thead>
<tr>
<th>EM Iteration</th>
<th>Proportions P [p1,p2]</th>
<th>Step 2: Multiply</th>
<th>Step 3: Add</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>p1 . X1</td>
<td>p2 . X2</td>
<td>Sum (p1 . X1 + p2 . X2)</td>
</tr>
<tr>
<td>1</td>
<td>[0.5000,0.5000]</td>
<td>[0.0039,0.0255,0.0033,0.0017]</td>
<td>[0.0015,0.0486,0.1085,0.0071]</td>
</tr>
<tr>
<td></td>
<td></td>
<td>R1 (Sum / p1 . X1)</td>
<td>R2 (Sum / p2 . X2)</td>
</tr>
<tr>
<td></td>
<td>[0.7141,0.3440,0.0299,0.1927]</td>
<td>0.3859,0.6560,0.9701,0.8073</td>
<td>[0.3202,0.6798]</td>
</tr>
<tr>
<td>2</td>
<td>[0.3202,0.6798]</td>
<td>[0.0025,0.0163,0.0021,0.0011]</td>
<td>[0.0021,0.0660,0.1475,0.0096]</td>
</tr>
<tr>
<td></td>
<td></td>
<td>R1 (Sum / p1 . X1)</td>
<td>R2 (Sum / p2 . X2)</td>
</tr>
<tr>
<td></td>
<td>[0.5405,0.1980,0.0143,0.0111]</td>
<td>0.4595,0.8020,0.9857,0.8989</td>
<td>[0.2135,0.7865]</td>
</tr>
<tr>
<td>3</td>
<td>[0.2135,0.7865]</td>
<td>[0.0017,0.0109,0.0014,0.0007]</td>
<td>[0.0024,0.0764,0.1707,0.0111]</td>
</tr>
<tr>
<td></td>
<td></td>
<td>R1 (Sum / p1 . X1)</td>
<td>R2 (Sum / p2 . X2)</td>
</tr>
<tr>
<td></td>
<td>[0.4040,0.1246,0.0083,0.0069]</td>
<td>0.5960,0.8754,0.9917,0.9391</td>
<td>[0.1494,0.8506]</td>
</tr>
<tr>
<td>4</td>
<td>[0.1494,0.8506]</td>
<td>[0.0012,0.0076,0.0010,0.0005]</td>
<td>[0.0026,0.0826,0.1846,0.0120]</td>
</tr>
<tr>
<td></td>
<td></td>
<td>R1 (Sum / p1 . X1)</td>
<td>R2 (Sum / p2 . X2)</td>
</tr>
<tr>
<td></td>
<td>[0.3050,0.0844,0.0054,0.0403]</td>
<td>0.6950,0.9156,0.9946,0.9597</td>
<td>[0.1088,0.8912]</td>
</tr>
</tbody>
</table>

Table 7: Execution Trace of Algorithm 1

Appendix B: Parallel Implementation

One of the approaches I tried to improve the computational efficiency of the land-use classification algorithm was to implement a parallelized solution. Details regarding the parallel implementation constitutes the remaining part of this report. Text used in remaining pages is actually taken from the conference paper accepted at the conference on Partitioned Global Address Space (PGAS) Programming Models, 2006.