Published in final edited form as:
Int J Hybrid Intell Syst. 2004 April 1; 1(1-2): 80–89.

A Framework for Learning from Distributed Data Using Sufficient Statistics and its Application to Learning Decision Trees

Doina Caragea, **Adrian Silvescu**, and **Vasant Honavar**
Artificial Intelligence Research Laboratory, Computer Science Department, Iowa State University, 226 Atanasoff Hall, Ames, IA 50011-1040, USA, dcaragea@cs.iastate.edu, silvescu@cs.iastate.edu, honavar@cs.iastate.edu

Abstract

This paper motivates and precisely formulates the problem of learning from distributed data; describes a general strategy for transforming traditional machine learning algorithms into algorithms for learning from distributed data; demonstrates the application of this strategy to devise algorithms for decision tree induction from distributed data; and identifies the conditions under which the algorithms in the distributed setting are superior to their centralized counterparts in terms of time and communication complexity; The resulting algorithms are *provably exact* in that the decision tree constructed from distributed data is identical to that obtained in the centralized setting. Some natural extensions leading to algorithms for learning from heterogeneous distributed data and learning under privacy constraints are outlined.

Keywords

Distributed Learning; Sufficient Statistics; Learning Agents; Decision Trees

1 Introduction

Recent advances in computing, communications, and digital storage technologies, together with development of high throughput data acquisition technologies have made it possible to gather and store large volumes of data in digital form. For example, advances in high throughput sequencing and other data acquisition technologies have resulted in gigabytes of DNA, protein sequence data, and gene expression data being gathered at steadily increasing rates in biological sciences; organizations have begun to capture and store a variety of data about various aspects of their operations (e.g., products, customers, and transactions); complex distributed systems (e.g., computer systems, communication networks, power systems) are equipped with sensors and measurement devices that gather and store a variety of data for use in monitoring, controlling, and improving the operation of such systems. These developments have resulted in unprecedented opportunities for large-scale data-driven knowledge acquisition with the potential for fundamental gains in scientific understanding (e.g., characterization of macromolecular structure-function relationships in biology) in many data-rich domains. In such applications, the data sources of interest are typically physically distributed and are often autonomous. Given the large size of these data sets, gathering all of the data in a centralized location is generally neither desirable nor feasible because of bandwidth and storage requirements. In such domains, there is a need for knowledge acquisition systems that can perform the necessary analysis of data at the locations where the data and the computational resources are available and transmit the results of analysis (knowledge acquired from the data) to the locations where they are needed (Honavar et al. 1998). In other domains, the ability of autonomous organizations to share raw data may be limited due to a variety of reasons (e.g., privacy considerations). In such cases, there is a need for knowledge acquisition algorithms that can learn from statistical summaries of data (e.g., counts of instances that match certain criteria) that are made available as needed from the distributed data sources in the absence of access to raw data.

Against this background, this paper presents an approach to the design of systems for learning from distributed, autonomous data sources. We precisely formulate a class of distributed learning problems; present a general strategy for transforming traditional machine learning algorithms into distributed learning algorithms; and demonstrate the application of this strategy to devise algorithms for decision tree induction (using a variety of splitting criteria) from distributed data. The resulting algorithms are *provably exact* in that the decision tree constructed from distributed data is identical to that obtained by the corresponding algorithms in the centralized setting (i.e., when all of the data is available in a central location) and they compare favorable to their centralized counterparts in terms of time and communication complexity.

The rest of the paper is organized as follows: Section 2 presents a precise formulation of the problem of learning from distributed data; Section 3 introduces a general strategy for transforming *centralized* learning algorithms into algorithms for learning from distributed data; Section 4 demonstrates an application of this strategy to devise algorithms for decision tree induction (using a variety of commonly used splitting criteria) from *horizontally* or *vertically* fragmented distributed data, and provides an analysis of the time and communication complexity of the proposed algorithms; Section 5 concludes with a discussion of related research and a brief outline of future research directions.

2 Learning from Distributed Data

The problem of learning from data can be summarized as follows: Given a data set D, a hypothesis class H, and a performance criterion P, the learning algorithm L outputs a hypothesis $h \in H$ that optimizes P. In pattern classification applications, h is a classifier (e.g., a decision tree, a support vector machine, etc.). The data D typically consists of a set of training examples. Each training example is an ordered tuple of attribute values, where one of the attributes corresponds to a class label and the remaining attributes represent inputs to the classifier. The goal of learning is to produce a hypothesis that optimizes the performance criterion of minimizing some function of the classification error (on the training data) and the complexity of the hypothesis. Under appropriate assumptions, this is likely to result in a classifier that assigns correct labels to unlabeled instances.

In a distributed setting, each data source contains only a *fragment* of the data set. Two common types of data fragmentation are: *horizontal fragmentation* (Figure 1 (Left)), wherein (possibly overlapping) subsets of data tuples are stored at different sites; and *vertical fragmentation* (Figure 1 (Right)), wherein (possibly overlapping) subtuples of data tuples are stored at different sites. More generally, the data may be fragmented into a set of *relations* (as in the case of tables of a relational database, but distributed across multiple sites). If a data set D is distributed among the sites $1, \cdots, n$ containing data set fragments $D_1, \cdots, D_n$, we assume that the individual data sets $D_1, \cdots, D_n$ collectively contain all the information needed to construct the complete dataset D (at least in principle).

The distributed setting typically imposes a set of constraints Z on the learner that are absent in the centralized setting. For example, the constraints Z may prohibit the transfer of raw data from each of the sites to a central location while allowing the learner to obtain certain statistics from the individual sites (e.g., counts of instances that have specified values for some subset of attributes). In some applications of data mining (e.g., knowledge discovery from clinical records), Z might include constraints designed to preserve privacy.

The problem of learning from distributed data can be summarized as follows: Given the fragments $D_1, \cdots, D_n$ of a data set D distributed across the sites $1, \cdots, n$, a set of constraints Z, a hypothesis class H, and a performance criterion P, the task of the learner L_d is to output a hypothesis $h \in H$ that optimizes P using only operations allowed by Z. Clearly, the problem of learning from a centralized data set D is a special case of learning from distributed data where $n = 1$ and $Z = \varphi$.

Having defined the problem of learning from distributed data, we proceed to define some criteria that can be used to evaluate the quality of the hypothesis produced by an algorithm L_d for learning from distributed data relative to its centralized counterpart.

We say that an algorithm L_d for learning from distributed data sets $D_1, \cdots, D_n$ is *exact* relative to its centralized counterpart L if the hypothesis produced by L_d is identical to that obtained by L from the complete data set D obtained by appropriately combining the data sets $D_1, \cdots, D_n$.

Example: Let L_d be an algorithm for learning a Support Vector Machine (SVM) classifier $h_d : \Re^n \rightarrow \{-1, 1\}$, under constraints Z, from horizontally fragmented distributed data $D_1, \cdots, D_n$, where each $D_i \subseteq \Re^n \times \{-1, 1\}$. Let L be a centralized algorithm for learning an SVM classifier $h : \Re^n \rightarrow \{-1, 1\}$ from data $D \subseteq \Re^n \times \{-1, 1\}$. If $D = \cup_1^n D_i$, then we say that L_d is exact with respect to L if and only if $\forall X \in \Re^n$, $h(X) = h_d(X)$.

Proof of exactness of an algorithm for learning from distributed data relative to its centralized counterpart ensures that a large collection of existing theoretical (e.g., sample complexity, error bounds) as well as empirical results obtained in the centralized setting apply in the distributed setting.

Similarly, we can define exactness of learning from distributed data with respect to other criteria of interest (e.g., expected accuracy of the learned hypothesis). More generally, it might be useful to consider *approximate* distributed learning in similar settings. However, we focus on algorithms for learning from distributed data that are provably *exact* with respect to their centralized counterparts in the sense defined above.

3 A General Strategy for Designing Algorithms for Learning from Distributed Data

Our general strategy for designing an algorithm for learning from distributed data that is provably exact with respect to its centralized counterpart (in the sense defined above) follows from the observation that most of the learning algorithms use only certain *statistics* computed from the data D in the process of generating the hypotheses that they output. (Recall that a *statistic* is simply a function of the data. Examples of statistics include mean value of an attribute, counts of instances that have specified values for some subset of attributes, the most frequent value of an attribute, etc.) This yields a natural decomposition of a learning algorithm into two components (See Figure 2): (1) an information extraction component that formulates and sends a *statistical query* to a data source and (2) a hypothesis generation component that uses the resulting statistic to modify a partially constructed hypothesis (and further invokes the information extraction component if needed).

A statistic $s(D)$ is called a *sufficient statistic* for a parameter θ if $s(D)$ (loosely speaking) provides all the information needed for estimating the parameter θ from data D. Thus, sample mean is a sufficient statistic for mean of a Gaussian distribution. A sufficient statistic s for a parameter θ is called a *minimal sufficient statistic* if for every sufficient statistic s_θ for θ there exists a function g_{s_θ} such that $g_{s_\theta}(s_\theta(D)) = s_\theta(D)) =$ (Casella and Berger 2001).

We can generalize this notion of a sufficient statistic for a parameter θ to yield a notion of a sufficient statistic $s_{L;h}(D)$ for learning a hypothesis h using a learning algorithm L applied to a data set D. Trivially, the data D is a sufficient statistic for learning h using L. However, we are typically interested in statistics that are minimal or at the very least, substantially smaller in size than the whole data set D.

In some simple cases, it is possible to extract a sufficient statistic $s_{L,h}(D)$ for constructing a hypothesis h in one step (e.g., when L is the standard algorithm for learning a Naive Bayes classifier). In such a case, we say that $s_{L,h}$ is a sufficient statistic for learning h using the learning algorithm L if there exists an algorithm that accepts $s_{L,h}(D)$ as input and outputs $h = L(D)$.

In practice, h is constructed by L by interleaving information extraction and hypothesis generation operations (see Figure 2). Thus, a decision tree learning algorithm would first obtain the sufficient statistics (expected information concerning the class membership of an instance associated with each of the attributes) for a single node decision tree (a partial hypothesis h_1), then follow up with queries for additional statistics needed to iteratively refine h_1 to obtain a succession of partial hypotheses h_1, $h_2 \cdots$ culminating in h.

We say that $s(D, h_i)$ is a sufficient statistic for the *refinement* of a hypothesis h_i into h_{i+1} (denoted by $s_{h_i \rightarrow h_{i+1}}$) if there exists an algorithm R which accepts h_i and $s(D, h_i)$ as inputs and outputs h_{i+1}.

We say that $s_h(D, h_1 \cdots h_m)$ is a sufficient statistic for the composition of the hypotheses $(h_1 \cdots h_m)$ into h (denoted by $s_{(h_1,\cdots,h_m)\rightarrow h}$) if there exists an algorithm C which accepts as inputs $h_1, \cdots, h_m$ and $s_h(D, h_1, \cdots, h_m)$ and outputs the hypothesis h.

We say that $s_{h_i \rightarrow h_{i+k}}$ (where $k \geq 0$) is a sufficient statistic for iteratively refining a hypothesis h_i into h_{i+k} if h_{i+k} can be obtained through a sequence of refinements starting with h_i. We say that $s_{(h_1 \cdots h_m)\rightarrow h}$ is a sufficient statistic for obtaining hypothesis h starting with hypotheses $h_1, \cdots, h_m$ if h can be obtained from $h_1 \cdots h_m$ through some sequence of applications of composition and refinement operations.

Assuming that the relevant sufficient statistics (and the procedures for computing them) can be defined, the application of a learning algorithm L to a data set D can be reduced to the computation of $s_{h_0 \rightarrow h}$ through some sequence of applications of hypothesis refinement and composition operations starting with the hypothesis $h_0 = \varphi$ (see Figure 2).

In light of the previous discussion, the task of designing an algorithm L_d for learning from distributed data can be decomposed into two components: (1) information extraction from distributed data and (2) hypothesis generation. Information extraction from distributed data entails decomposing each statistical query q posed by the information extraction component of the learner into subqueries $q_1, \cdots, q_n$ that can be answered by the individual data sources $D_1, \cdots, D_n$, respectively, and a procedure for combining the answers to the subqueries into an answer for the original query q (See Figure 3). When the learner's access to data sources is subject to constraints Z the resulting plan for information extraction has to be executable without violating the constraints Z. The *exactness* of the algorithm L_d for learning from distributed data relative to its centralized counterpart, which requires access to the complete data set D, follows from the correctness (soundness) of the query decomposition and answer composition procedure.

The transformation of the task of learning from distributed data into a sequence of applications of hypothesis refinement and hypothesis composition operations can be performed assuming serial or parallel access to the data sources $D_1, \cdots, D_n$ (see Figure 4).

In the next section, we will demonstrate the application of the general approach described above in the case of learning decision trees from distributed data.

4 Decision Tree Induction from Distributed Data

Decision tree algorithms (Quinlan 1986, Breiman et al. 1984) are among some of the most widely used machine learning algorithms for building pattern classifiers from data. Their popularity is due in part to their ability to: select from all attributes used to describe the data, a subset of attributes that are relevant for classification; identify complex predictive relations among attributes; and produce classifiers that are easy to comprehend for humans.

The ID3 (Iterative Dichotomizer 3) algorithm proposed by Quinlan (Quinlan 1986) and its more recent variants represent a widely used family of decision tree learning algorithms. The ID3 algorithm searches in a greedy fashion, for attributes that yield the maximum amount of information for determining the class membership of instances in a training set D of labeled instances. The result is a decision tree that correctly assigns each instance in D to its respective class. The construction of the decision tree is accomplished by recursively partitioning D into subsets based on values of the chosen attribute until each resulting subset has instances that belong to exactly one of the m classes. The selection of an attribute at each stage of construction of the decision tree maximizes the estimated expected information gained from knowing the value of the attribute in question.

Consider a set of instances S which is partitioned into m disjoint subsets (classes) $C_1, C_2, \ldots, C_m$ such that $D=\bigcup_{i=1}^{M} C_i$ and $C_i \cap C_j = \varnothing \ \forall i \neq j$. The estimated probability that a randomly chosen instance $x \in D$ belongs to the class C_j is $p_j=\frac{|C_j|}{|D|}$, where $|X|$ denotes the cardinality of the set X. The estimated *entropy* of a set D measures the expected information needed to identify the class membership of instances in D, and is defined as follows: $entropy(D)=-\sum_j \frac{|C_j|}{|D|} \cdot \log_2\left(\frac{|C_j|}{|D|}\right)$. Given some impurity measure, the entropy (Quinlan 1986) or Gini index (Breiman et al. 1984), or any other measure that can be defined based on the probabilities p_j, we can define the estimated *information gain* for an attribute a, relative to a collection of instances S as follows: $I\,Gain(D,a)=I(D)-\sum_{v\in Values(a)} \frac{|D_v|}{|D|} I(D_v)$, where $Values(A)$ is the set of all possible values for attribute a, D_v is the subset of D for which attribute a has value v, and $I(D)$ can be *entropy*(D), Gini index, or any other suitable measure.

Thus, the information requirements of ID3-like decision tree learning algorithms can be expressed in terms of relative frequencies computed from the relevant instances at each node. These relative frequencies represent refinement sufficient statistics in the sense defined in the previous section. Different algorithms for decision tree induction differ from each other in terms of the criterion that is used to evaluate the splits that correspond to tests on different candidate attributes. The choice of the attribute at each node of the decision tree greedily maximizes (or minimizes) the chosen splitting criterion (Caragea et al. 2003).

To keep things simple, we assume that all the attributes are discrete or categorical. However, all the discussion below can be easily generalized to continuous attributes (Witten, I. and Frank, E. 1999). Often, decision tree algorithms also include a pruning phase to alleviate the problem of overfitting the training data. For the sake of simplicity of exposition, we limit our discussion to decision tree construction without pruning. However, it is relatively straightforward to modify the proposed algorithms to incorporate a variety of pruning methods.

4.1 Statistics Gathering from Distributed Data

Assume that given a partially constructed decision tree, we want to choose the best attribute for the next split. Let $a_j(\pi)$ denote the attribute at the jth node along a path π starting from the attribute $a_1(\pi)$ that corresponds to the root of the decision tree, leading up to the node in question $a_l(\pi)$ at depth l. Let $v(a_j(\pi))$ denote the value of the attribute $a_j(\pi)$, corresponding to the jth node along the path π. For adding a node below $a_l(\pi)$, the set of examples being considered satisfy the following constraints on values of attributes: $L(\pi) = [a_1(\pi) = v(a_1(\pi))] \wedge [a_2(\pi) = v(a_2(\pi))] \cdots [a_l(\pi) = v(a_l(\pi))]$ where $[a_j(\pi) = v(a_j(\pi))]$ denotes the fact that the value of the jth attribute along the path π is $v(a_j(\pi))$.

It follows from the preceding discussion that the sufficient statistics for constructing decision trees are the counts of examples that satisfy specified constraints on the values of particular attributes. These counts have to be obtained once for each node that is added to the tree starting with the root node. If we can devise distributed statistics gathering operators for obtaining the necessary counts from distributed data sets, we can obtain exact distributed decision tree learning algorithms. Thus, the decision tree constructed from a given data set in the distributed setting is exactly the same as that obtained in the batch setting when using the same splitting criterion in both cases.

4.1.1 Horizontally Fragmented Distributed Data—

When the data is horizontally distributed, examples corresponding to a particular value of a particular attribute are scattered at different locations. In order to identify the best split at a particular node in a partially constructed tree, all the sites are visited and the counts corresponding to candidate splits of that node are accumulated. The learner uses these counts to find the attribute that yields the best split to further partition the set of examples at that node. Thus, given $L(\pi)$, in order to split the node corresponding to $a_l(\pi) = v(a_l(\pi))$, the statistics gathering component has to obtain the counts of examples that belong to each class for each possible value of each candidate attribute.

Let $|D|$ be the total number of examples in the distributed data sets; $|A|$, the number of attributes; V the maximum number of possible values per attribute; n the number of sites; m the number of classes; and $size(T)$ the number of nodes in the decision tree. For each node in the decision tree T, the statistics gathering component has to scan the data at each site to calculate the corresponding counts. We have: $\sum_{i=1}^{n} |D_i| = |D|$. Therefore, in the case of serial access to the distributed data sources (Figure 4 (Left)), the time complexity of the resulting algorithm is $O(|D||A| \cdot size(T))$. In the case of parallel access to the data sources (Figure 4 (Right)), this can be further improved since each site can perform information extraction in parallel. For each node in the decision tree T, each site has to transmit the counts based on its local data. These counts form a matrix of size $m|A|V$. Hence, the communication complexity (the total amount of information that is transmitted between sites) is given by $O(m|A||V|n \cdot size(T))$. It is worth noting that some of the bounds presented here can be further improved so that they depend on the height of the tree instead of the number of nodes in the tree.

4.1.2 Vertically Fragmented Distributed Data—

In vertically distributed datasets, we assume that each example has a unique index associated with it. Subtuples of an example are distributed across different sites. However, correspondence between subtuples of a tuple can be established using the unique index. As before, given $L(\pi)$, in order to split the node corresponding to $a_l(\pi) = v(a_l(\pi))$, the statistics gathering component has to obtain the counts of examples that belong to each class for each possible value of each candidate attribute. Since each site has only a subset of the attributes, the set of indices corresponding to the examples that match the constraint $L(\pi)$ have to be transmitted to the sites. Using this information, each site can compute the relevant counts that correspond to the attributes that are stored at the site. The hypothesis generation component uses the counts from all the sites to select the attribute to further split the node corresponding to $a_l(\pi) = v(a_l(\pi))$. For each node in the decision tree T, each site has to compute the relevant counts of examples that satisfy $L(\pi)$ for the attributes stored at that site. The number of subtuples stored at each site is $|D|$ and the number of attributes at each site is bounded by the total number of attributes $|A|$. In the case of serial access to distributed data sources, the time complexity is given by $O(|D||A|n \cdot size(T))$. In the case of parallel distributed learning since the various sites can perform information extraction in parallel, this can be further improved. For each node in the tree T, we need to transmit to each site, the set of indices for the examples that satisfy corresponding constraint $L(\pi)$ and get back the relevant counts for the attributes that are stored at that site. The number of indices is bounded by $|D|$ and the number of counts is bounded by $m|A|V$. Hence, the communication complexity is given by $O((|D| + m|A|V)n \cdot size(T))$. Again, it is possible to further improve some of these bounds so that they depend on the height of the tree instead of the number of nodes in the tree.

4.2 Algorithm for Learning Decision Trees from Distributed Data Compared with its Centralized Counterpart

Our approach to learning decision trees from distributed data based on the decomposition of the learning task into a distributed statistics gathering component and a hypothesis generation component provides an effective way to deal with scenarios in which the sites provide only statistical summaries of the data on demand and prohibit access to raw data. Even when it is possible to access the raw data, the distributed algorithm compares favorably with its centralized counterpart which needs access to the entire data set, whenever its communication cost is less than the cost of collecting all of the data in a central location. It follows from the preceding analysis that in the case of horizontally fragmented data, the distributed algorithm has an advantage when $mV\, n \cdot size(T) \leq |D|$ since the cost of shipping the data is given by its actual size, which is given by $|D||A|$. In the case of vertically fragmented data, the corresponding conditions are given by $size(T) \leq |A|$ since the cost of shipping the data is given by its actual size, which has a lower bound of $|D||A|$. These conditions are often met in the case of large, high-dimensional data sets.

5 Summary and Discussion

5.1 Summary

Efficient learning algorithms with provable performance guarantees for learning from from distributed data constitute a key element of any practical approaches to data driven discovery and decision making using large, autonomous data repositories that are becoming available in many domains (e.g., biological sciences, atmospheric sciences). In this paper, we have precisely formulated a class of distributed learning problems and described a general strategy for transforming standard machine learning algorithms that assume centralized access to data in a single location into algorithms for learning from distributed data. We have demonstrated the application of this strategy to devise algorithms for decision tree induction from distributed data. The resulting algorithms are *provably exact* in that the decision tree constructed from distributed data is identical to that obtained by the corresponding algorithm when it is used in the centralized setting. This ensures that the entire body of theoretical (e.g., sample complexity, error bounds) and empirical results obtained in the centralized setting carry over to the distributed setting. We have also identified the conditions under which it is advantageous to use the algorithm for learning from distributed data instead of its centralized counterpart.

5.2 Discussion

Distributed learning has received considerable attention in the literature. Some of the distributed learning algorithms proposed focus on distributing a large centralized data set to multiple processors to exploit parallel processing to speed up learning. In contrast, the approach proposed in this paper focuses on learning from a set of autonomous distributed data sources.

The autonomous nature of the data sources implies that the learner has no control over the manner in which the data is distributed across the different sources.

Other algorithms for learning classifiers from distributed data (Domingos 1997, Prodromidis et al. 2000) learn seperate hypothesis from each of the data sets and combine them (typically using a weighted voting scheme) to obtain an ensemble classifier. This typically requires transmission of a subset of data from each of the data sources to the central site for use in determining the weights to be assigned to the individual hypotheses, precluding their use when privacy constraints forbid such data transmission. In contrast, the approach described in this paper is applicable in scenarios which preclude transmission of data but allow transmission of minimal sufficient statistics needed by the learning algorithm. Another important limitation of the ensemble classifier approach to learning from distributed data is the lack of guarantees concening generalization accuracy of the resulting hypothesis relative to the hypothesis obtained in the centralized setting.

Most of the algorithms proposed in the literature for learning classifiers from distributed data, barring a few exceptions (Kargupta et al. 1999, Bhatnagar and Srinivasan 1997), assume horizontal fragmentation. Kargupta et al (Kargupta et al. 1999) describe an algorithm for learning decision trees from vertically fragmented data. The authors make use of the fact that a decision tree can be viewed as a boolean function that can be approximated using only low order Fourier coefficients (coefficients corresponding to attribute combinations whose size is at most logaritmic in the number of nodes in the tree) (Mansour 1994). At each local site, the learner estimates the Fourier coefficients from the local data, and transmits them to a central site. These estimates are combined to obtain a set of Fourier coefficients for the decision tree (the transmission of a subset of the data from each local siteto the central site is needed to compute some coefficients that can not be computed locally). At present, there are no guarantees of performance of the hypothesis obtained in the distributed setting relative to that obtained in the centralized setting. An added complication has to do with the fact that a set of Fourier coefficients can correspond to multiple decision trees. The algorithm proposed in (Bhatnagar and Srinivasan 1997), is similar to our algorithm for learning decision trees from vertically fragmented data.

The approach described in this paper for learning classifiers from distributed data guarantees that the hypothesis obtained in the distributed setting is provably identical to that obtained in the centralized setting. It also opens up the possibility for considering variants of the proposed approach that yield bounded error approximations in the distributed setting of the hypothesis produced by centralized counterparts under resource (computation, memory, bandwidth) constraints.

Our approach to learning from distributed data relies on a decomposition of the learning task into two components: extraction of sufficient statistics from data and hypothesis generation. The particular statistics that are extracted from the data depend on the *structure* of the hypothesis. In the case of decision trees, the form of the hypothesis corresponds to a Boolean expression that is a disjuction of conjunctions (where each conjunction corresponds to a path from the root to a leaf of the decision tree). Thus, the structure of the hypothesis can be described by a symbolic expression. What is perhaps not so obvious is that the decision tree also has a set of *parameters* associated with it, namely, the class distribution of examples at the leaves of the tree. The structure of the tree, along with these parameters, in fact constitutes a probabilistic classifier. Thus, decision trees can in fact be viewed as examples of *hybrid* representations which include a symbolic or structual part and a set of numeric parameters (e.g., probabilities). Bayesian networks, Hidden Markov Models, and other hypothesis classes are also examples of such hybrid representations.

It is important to note that the general strategy for learning classifiers from distributed data is applicable to the entire class of algorithms for learning classifiers from data. This follows from the fact that the output *h* of any learning algorithm is in fact a function of the data *D*, and hence by definition, a *statistic*. Consequently, we can devise a strategy for computing *h* from the data *D* through some combination of refinement and composition operations starting with an initial hypotheses (or an initial set of hypotheses). The seperation of concerns between hypothesis construction and extraction of sufficient statistics from data makes it possible to explore the use of sophisticated techniques for query optimization that yield *optimal* plans for gathering sufficient statistics from distributed data sources under a specified set of constraints that describe the query capabilities and operations permitted by the data sources (e.g., execution of user supplied procedures).

This makes the proposed approach to learning from distributed data applicable in a broad range of data fragmentation scenarios. Of particular interest is learning from distributed data sources when the ontologies (names for terms, relationships among terms) associated with the individual data sources are different from each other. Provided well-defined mappings between ontologies can be specified, the proposed approach to learning from distributed data can be extended yield an approach to learning from *heterogeneous* distributed data of the sort encountered in many large scale scientific applications (Reinoso-Castillo et al. 2003).

5.3 Future Work

The design of algorithms for learning from distributed data described in this paper has been motivated by the desirability of performing as much of the processing of data as feasible at the sites where the data and computing resources are available to avoid retrieving large volumes of data from remote sites. The applicability of the proposed approach in practice depends on whether information requirements of the centralized learning algorithm *L* under consideration can be met under the constraints imposed by the distributed setting and the time, memory, and communication costs of the resulting algorithm relative to the other alternatives (e.g., gathering all of the data in a centralized site and then applying the centralized learning algorithm if such a solution is allowed by the constraints *Z*). In general, it is desirable to use a query optimization procedure to decompose a statistical query *q* to generate an optimal plan of operation where the primitive operations correspond to queries that can be directly executed on the individual data sources based on the relative costs of the alternatives available. Query optimization in settings where some of the operations that need to be performed on data from remote sites can be executed by code shipped to remote sites has been investigated in (Rodriguez-Martinez and Roussopoulos 2000). It is of interest to incorporate similar techniques for query optimization in the statistical query decomposition component of the proposed algorithms for learning from distributed data for different choices of constraints *Z* (e.g., privacy constraints in knowledge acquisition from clinical records) that arise in practice.

The discussion in this paper has focused primarily on algorithms for learning from distributed data that are provably exact relative to their centralized counterparts. In many applications, it would be of interest to relax the exactness requirement leading to provably approximate algorithms (based on resource constrained approximations of sufficient statistics). Also of interest are extensions of the proposed approach to *cumulative* and *incremental* learning scenarios (Caragea et al. 2001, Polikar et al. 2001) as well as collaborative learning in communities of multiple autonomous agents (in which each agent has its own private knowledge and goals and access to some limited set of information sources and can benefit from interaction with other agents).

Many distributed data sources are also heterogeneous in structure (e.g., relational databases, information systems accessed through web interfaces that support a limited set of queries, image databases, text databases, databases of molecular sequences, 3-dimensional structures), semantics (because of differences in underlying choices of real world entities and relationships being modeled and the terms used to denote them), granularity of information, and query capabilities. The approach to learning from distributed data proposed in this paper is motivated by the considerations that arise in learning from heterogeneous data sources. As noted above, it is possible to generate plans for obtaining sufficient statistics from heterogeneous data sources when mappings between ontologies are specified. Related work in our laboratory has focused on the development of the INDUS (Intelligent Data Understanding System) architecture for data-driven knowledge acquisition from heterogeneous, distributed information sources. The information integration component of INDUS allows users to define customized views to integrate data from remote sources (Reinoso-Castillo et al. 2003). Work in progress is aimed at extending INDUS to support execution of queries for sufficient statistics needed by the learning algorithms (expressed in terms of the learner's ontology) over heterogeneous distributed data sources. This would allow us to extend the algorithms proposed in this paper to work with heterogeneous distributed data and to perform case studies and extensive comparisons of performance of the resulting algorithms with their centralized counterparts.

It is also of interest to extend algorithms for learning from attribute value taxonomies and data (Zhang, J. and Honavar, V. 2003) to distributed settings, where data at different sites may be specified in terms of attribute values at different levels of granularity (corresponding to different levels of an attribute value taxonomy). In related work, we have developed algorithms for learning from multiple tables in a relational database (Atramentov et al. 2003). It is of interest to explore approaches similar to those described in this paper for learning from distributed relational databases as well as heterogeneous distributed data which are presented by INDUS as if they were set of relations. Some of the work in progress is aimed at application of the proposed algorithms to knowledge acquisition tasks that arise in applications in computational biology, medical informatics, information security, and related domains.

Acknowledgments

This work has been supported in part by grants from the National Science Foundation (IIS 0219699), and the National Institutes of Health (GM 066387) to Vasant Honavar and an IBM Ph.D. fellowship to Doina Caragea.

References

Atramentov, A.; Leiva, H.; Honavar, V. Learning Decision Trees from Multi-Relational Data. In: Horvth, T.; Yamamoto, A., editors. Proceedings of the 13th International Conference on Inductive Logic Programming, vol. 2835 of Lecture Notes in Artificial Intelligence; Springer-Verlag; 2003. p. 38-56.

Bhatnagar, R.; Srinivasan, S. Pattern discovery in distributed databases. Proceedings of the Fourteenth AAAI; Providence, RI: AAAI Press/The MIT Press; 1997. p. 503-508.

Breiman, L.; Friedman, J.; Olshen, R.; Stone, C. Classification and Regression Trees. Wadsworth & Brooks; Monterey CA: 1984.

Caragea, D.; Silvescu, A.; Honavar, V. Decision Tree Induction from Distributed Heterogeneous Autonomous Data Sources. Proceedings of the International Conference on Intelligent Systems Design and Applications; Tulsa, Oklahoma. 2003. In press

Caragea, D.; Silvescu, A.; Honavar, V. Emerging Neural Architectures Based on Neuroscience. Berlin: Springer-Verlag; 2001. Towards a Theoretical Framework for Analysis and Synthesis of Agents That Learn from Distributed Dynamic Data Sources; p. 547-559.

Casella, G.; Berger, RL. Statistical Inference. Duxbury Press; Belmont, CA: 2001.

Domingos, P. Knowledge acquisition from examples via multiple models. Proceedings of the Fourteenth International Conference on Machine Learning; Nashville, TN: Morgan Kaufmann; 1997. p. 98-106.

Kargupta, H.; Park, BH.; Hershberger, D.; Johnson, E. Collective Data Mining: A New Perspective Toward Distributed Data Mining. In: Kargupta, H.; Chan, P., editors. Advances in Distributed and Parallel Knowledge Discovery. Cambridge, MA: MIT Press; 1999.

Honavar, V.; Miller, L.; Wong, J. Distributed Knowledge Networks. Proceedings of the IEEE Information Technology Conference; Syracuse, NY: IEEE Press; 1998.

Mansour, Y. Learning Boolean Functions via the Fourier Transform. In: Roychowdhury, VP.; Siu, KY.; Orlitsky, A., editors. Theoretical Advances in Neural Computation and Learning. Kluwer: 1994.

Polikar, R.; Udpa, L.; Udpa, S.; Honavar, V. IEEE Transactions on Systems, Man, and Cybernetics. Vol. 31. 2001. Learn++: An Incremental Learning Algorithm for Multi-Layer Perceptron Networks; p. 497-508.

Prodromidis, A.; Chan, P.; Stolfo, S. Meta-learning in distributed data mining systems: Issues and approaches. In: Kargupta, H.; Chan, P., editors. Advances of Distributed Data Mining. AAAI Press; 2000.

Quinlan R. Induction of decision trees. Machine Learning 1986;1:81–106.

Reinoso-Castillo, J.; Silvescu, A.; Caragea, D.; Pathak, J.; Honavar, V. Information Extraction and Integration from Heterogeneous, Distributed, Autonomous Information Sources: A Federated, Query-Centric Approach. IEEE International Conference on Information Integration and Reuse; 2003. In press

Rodriguez-Martinez, M.; Roussopoulos, R. MOCHA: A Self-Extensible Database Middleware System for Distributed Data Sources. Proceedings of the 2000 ACM SIGMOD International Conference on Management of Data; Dallas, TX. 2000. p. 213-224.

Witten, I.; Frank, E. Data mining: practical machine learning tools and techniques with Java implementations. Morgan Kaufmann; San Francisco, CA: 1999.

Zhang, J.; Honavar, V. Learning Decision Tree Classifiers from Attribute-Value Taxonomies and Partially Specified Data. In: Fawcett, T.; Mishra, N., editors. Proceedings of the International Conference on Machine Learning; Washington, DC: AAAI Press; 2003. p. 880-887.

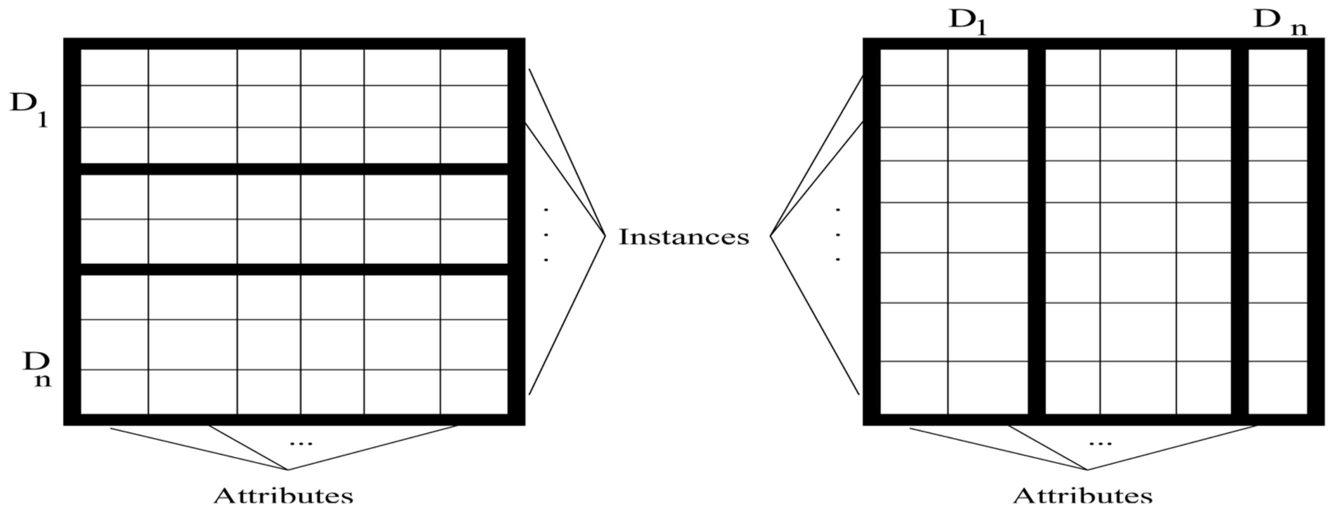

Figure 1.
Data Fragmentation: (Left) Horizontally Fragmented Data. (Right) Vertically Fragmented Data.

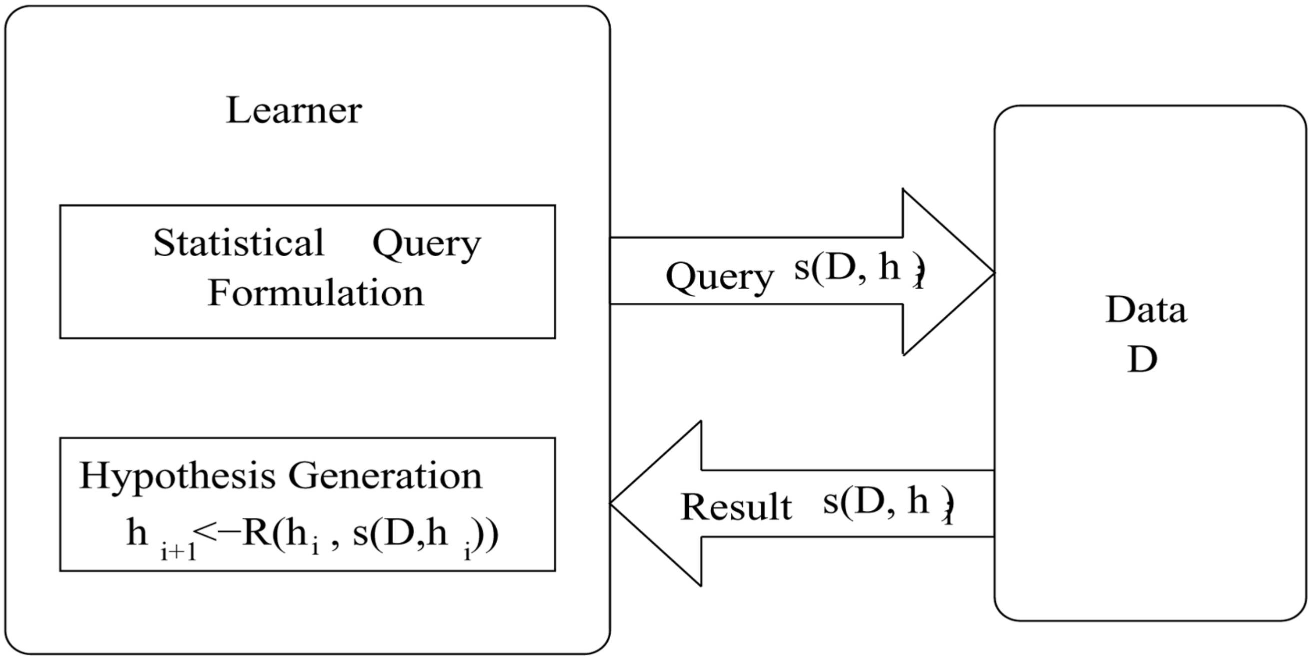

Figure 2.
Learning Revisited: identify sufficient statistics, gather the sufficient statistics and generate the current hypothesis.

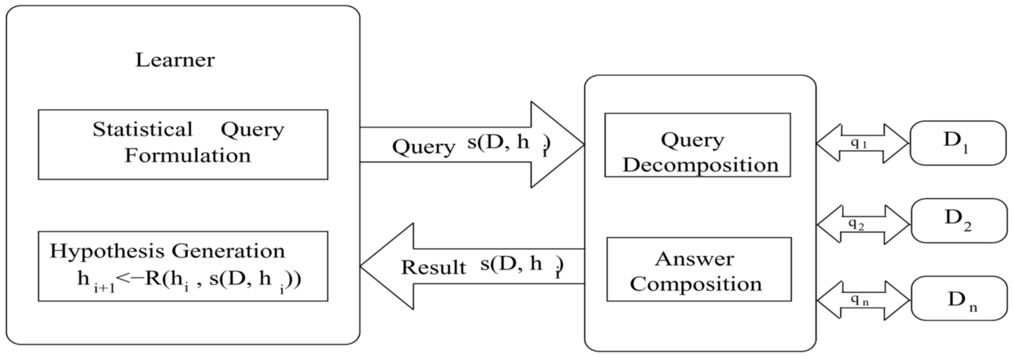

Figure 3.
Exact Distributed Learning: distribute the statistical query among the distributed data sets and compose their answers.

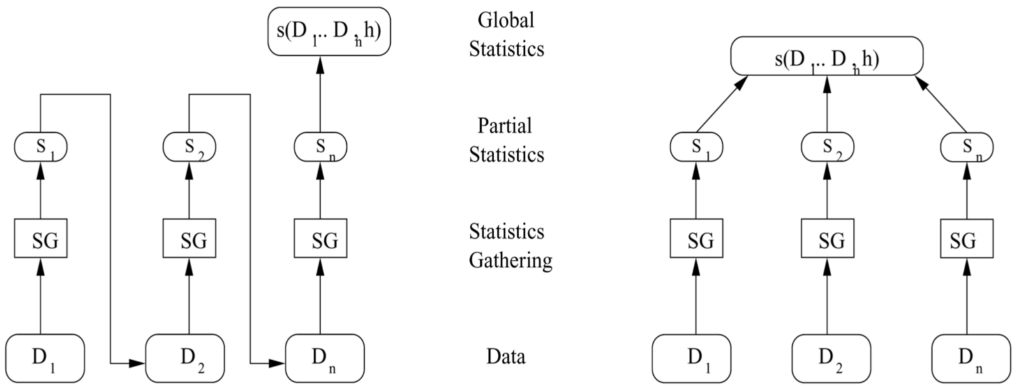

Figure 4.
Distributed Statistics Gathering: (Left) Serial. (Right) Parallel.