Learning Classifiers from Distributed Data Sources

Doina Caragea*
Department of Computing and Information Sciences
Kansas State University
234 Nichols Hall
Manhattan, KS 66506
USA
voice: +1 785-532-7908
fax: +1 785-532-7353
email: dcaragea@ksu.edu

Vasant Honavar
Department of Computer Science
Iowa State University
226 Atanasoff Hall
Ames, IA 50011
USA
voice: +1 515-294-0258
email: honavar@cs.iastate.edu

(* Corresponding author)

Learning Classifiers from Distributed Data Sources

Doina Caragea, Kansas State University, USA

Vasant Honavar, Iowa State University, USA

ABSTRACT

Many application domains have witnessed, in recent years, an exponential increase in the number, size and diversity of autonomous data sources. The large size and the distributed nature of the data, storage and bandwidth considerations, and in some instances, privacy considerations prevent centralized access to data that is typically assumed in standard machine learning approaches to knowledge acquisition. Hence, ransforming this explosive increase in data into commensurate increase in knowledge call for scalable and cost-effective approaches for building predictive models from distributed data. This chapter summarizes a sufficient-statistics based approach to learning a broad class of classifiers (including naïve Bayes and decision tree classifiers) from distributed data. This approach has been shown to yield results that are identical to those obtainable in settings where the learning algorithm has centralized access to all of the data. It can also be extended to settings where the distributed data sources differ from each other with respect to their structure (schema) and semantics e.g., choice of attribute names, values, and granularity of data descriptions.

INTRODUCTION

Recent development of high throughput data acquisition technologies in a number of domains (e.g., biological sciences, atmospheric sciences, space sciences, commerce) together with advances in digital storage, computing, and communications technologies have resulted in the

proliferation of a multitude of physically distributed data repositories created and maintained by autonomous entities (e.g., scientists, organizations). The resulting increasingly data rich domains offer unprecedented opportunities in computer assisted data-driven knowledge acquisition in a number of applications including in particular, data-driven scientific discovery (e.g., characterization of protein sequence-structure-function relationships in computational molecular biology), data-driven decision making in business and commerce, monitoring and control of complex systems (e.g., load forecasting in electric power networks), and security informatics (discovery of and countermeasures against attacks on critical information and communication infrastructures).

Machine learning (Mitchell, 1997; Duda et al., 2000), offers one of the most cost-effective approaches to analyzing, exploring and extracting knowledge (features, correlations, and other complex relationships and hypotheses that describe potentially interesting regularities) from data. However, the applicability of current approaches to machine learning in emerging data rich applications is severely limited by a number of factors:

a. Data repositories are large in size, dynamic, and physically distributed. Consequently, it is neither desirable nor feasible to gather all of the data in a centralized location for analysis. Hence, there is a need for efficient algorithms for analyzing and exploring multiple distributed data sources without transmitting large amounts of data.

b. Autonomously developed and operated data sources often differ in their structure and organization (e.g., relational databases, flat files, etc.) and the operations that can be performed on the data sources (e.g., types of queries - relational queries, statistical queries, keyword matches). Hence, there is a need for theoretically well-founded

strategies for efficiently obtaining the information needed for analysis within the operational constraints imposed by the data sources.

The purpose of this entry is to precisely define the problem of learning classifiers from distributed data and summarize recent advances that have led to a solution to this problem (Caragea et al., 2004; 2005).

BACKGROUND: PROBLEM SPECIFICATION

Given a data set D, a hypothesis class H, and a performance criterion P, an algorithm L for learning (from centralized data D) 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.) (See Figure 1). 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 (e.g., minimizing classification error on the training data) and the complexity of the hypothesis.

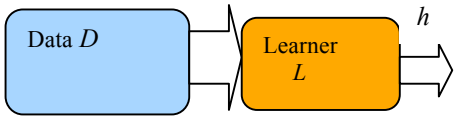

Figure 1: Learning from centralized data

In a distributed setting, a data set D is distributed among the sites 1,...,n containing the data set fragments D_1,...,D_n. Two common types of data fragmentation are*: horizontal fragmentation* and *vertical fragmentation*. In the case of horizontal fragmentation, each site contains a subset of the data tuples that make up D, i.e., $D = \bigcup_{i=1}^{n} D_i$. In the case of vertical fragmentation each site stores the subtuples of data tuples (corresponding to a subset of the attributes used to define data tuples

in D). In this case, D can be constructed by taking the *join* of the individual data sets $D_1,...,D_n$ (assuming a unique identifier for each data tuple is stored with the corresponding subtuples). 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) i.e., $D = \bigotimes_{i=1}^{n} D_i$ (where $\otimes$ denotes the *join* operation). If a data set D is distributed among the sites $1,\ldots,n$ containing data set fragments $D_1,..., D_n$, we assume that the individual data sets $D_1,...,D_n$ collectively contain (in principle) all the information needed to construct the dataset D. More generally, D may be fragmented across multiple relations (Ozsu & Valduriez, 1999; Friedman et al., 1999).

The distributed setting typically imposes a set of constraints Z on the learner. These constraints 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 types of 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 stated as follows (Caragea et al., 2004; 2005): Given the fragments $D_1,\ldots, D_n$ of a data set D distributed across the sites $1,...,n$, 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 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=∅.

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, ..., 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 data set D obtained by appropriately combining the data sets D_1,..., D_n.

The 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.

MAIN THRUST: STRATEGY FOR LEARNING FROM DISTRIBUTED DATA

Decomposition of the Learning from Data Task

A general strategy for designing algorithms for learning from distributed data that are provably exact with respect to their 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.).

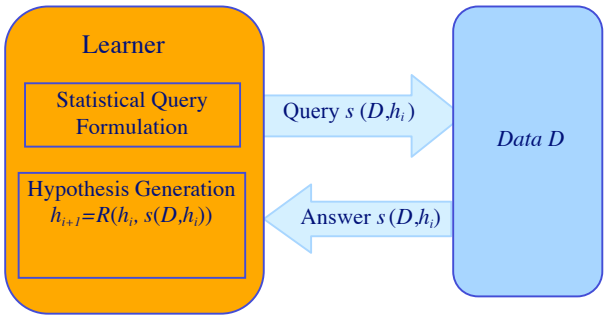

Figure 2: Learning = Statistical Query Answering & Hypothesis Generation

This yields a natural decomposition of a learning algorithm into two components (see Figure 2):

a. An information extraction component that formulates and sends a statistical query to a data source and

b. A hypothesis generation component that uses the resulting statistic to modify a partially constructed hypothesis (and it may further invoke the information extraction component as needed).

Sufficient Statistics for Learning

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 from data D. Thus, sample mean is a sufficient statistic for the 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 $g_{s_\theta}(s_\theta(D)) = s(D)$ (Casella & Berger, 2001).

This notion of a sufficient statistic for a parameter θ can be generalized to yield a sufficient statistic $s_{L,h}(D)$ for learning a hypothesis h using a learning algorithm L applied to a data set D (Caragea et al., 2004). 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 (in terms of the number of bits needed for encoding) than the 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., by querying the data source for a set of conditional probability estimates when L is the standard algorithm for learning a Naive Bayes classifier). In such a case, we say that $s_{L,h}(D)$ 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 a more general setting, h

is constructed by L by interleaving information extraction (statistical query) and hypothesis generation operations. Thus, a decision tree learning algorithm would start with an empty initial hypothesis h_0, obtain the sufficient statistics (expected information concerning the class membership of an instance associated with each of the attributes) for the root of the decision tree (a partial hypothesis h_1), and recursively generate queries for additional statistics needed to iteratively refine h_1 to obtain a succession of partial hypotheses $h_1,h_2,\ldots$ culminating in h (See Figure 2).

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 $s_{h_i \to h_{i+1}}$) 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 $s_h(D,h_1,\ldots h_m)$ is a sufficient statistic for the *composition* of the hypotheses $(h_1 \ldots h_m)$ into h (denoted by $s_{(h_1,\ldots h_m)\to h}$) if there exists an algorithm C which accepts as inputs $h_1,\ldots h_m$ and $s_h(D,h_1,\ldots h_m)$ and outputs the hypothesis h. We say that $s_{h_i \to h_{i+k}}$ (where $i \geq 0$ and $k > 0$ are positive integers) 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 $s_{(h_1,\ldots h_m)\to h}$ is a sufficient statistic for obtaining hypothesis h starting with hypotheses $h_1,\ldots h_m$ if h can be obtained from $h_1,\ldots h_m$ 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 $s_{(h_0,\ldots h_m)\to h}$ through some sequence of applications of hypothesis refinement and composition operations starting with the hypothesis h (See Figure 2). In this model, the only interaction learner's interaction with the repository of data D is through queries for the relevant statistics.

Information Extraction from Distributed Data

Based on the decomposition of the learning task into information extraction and hypothesis generation, learning from distributed data reduces to information extraction from distributed data. The information extraction from distributed data entails decomposing each statistical query q posed by the information extraction component of the learner into sub queries $q_1,...q_n$ that can be answered by the individual data sources $D_1,..,D_n$ respectively, and a procedure for combining the answers to the sub queries into an answer for the original query q. (See Figure 3).

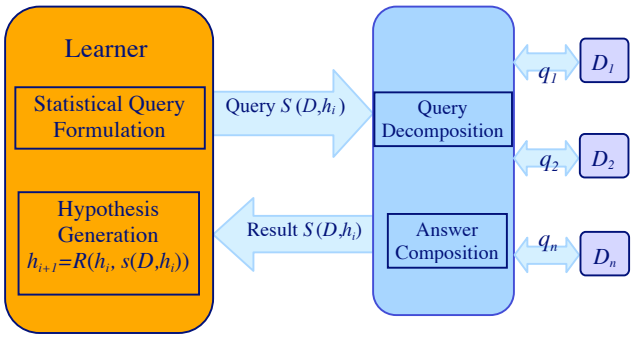

Figure 3: Learning from Distributed Data = Statistical Query Answering + Hypothesis

It is important to note that this 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 hypothesis (or an initial set of hypotheses). When the data D is stored in one or more relational database(s), this approach is able to effectively use the database support for aggregate queries for efficient data mining. 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 separation 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. These techniques yield optimal plans for gathering sufficient statistics from distributed data sources under a specified set of constraints. The constraints describe the query capabilities of the data sources, operations permitted by the data sources (e.g., execution of user supplied procedures), and available computation, bandwidth, and memory resources

Related Work

Srivastava et al. (1999) propose methods for distributing a large centralized data set to multiple processors to exploit parallel processing to speed up learning. Grossman and Guo (2001), and Provost and Kolluri (1999) survey several methods that exploit parallel processing for scaling up data mining algorithms to work with large data sets. In contrast, the focus of the proposed research is on learning classifiers from a set of autonomous distributed data sources. The autonomous nature of the data sources implies that the learner has little control over the manner in which the data are distributed among the different sources.

Distributed data mining has received considerable attention in the literature (Park, & Kargupta, 2002). Domingos (1997) and Prodromidis et al. (2000) propose an *ensemble of classifiers* approach to learning from horizontally fragmented distributed data which essentially involves learning separate classifiers from each data set and combining them typically using a weighted voting scheme. This typically requires gathering a subset of data from each of the data sources at a central site to determine the weights to be assigned to the individual hypotheses (or alternatively shipping the ensemble of classifiers and associated weights to the individual data sources where they can be executed on local data to set the weights). In contrast, the approach

described here is applicable even in scenarios that preclude transmission of data or execution of user-supplied code at the individual data sources, but allow transmission of minimal sufficient statistics needed by the learning algorithm. A second potential drawback of the ensemble of classifiers approach to learning from distributed data is that the resulting ensemble of classifiers is typically much harder to comprehend than a single classifier. A third important limitation of the ensemble classifier approach to learning from distributed data is the lack of guarantees concerning generalization accuracy of the resulting hypothesis relative to the hypothesis obtained in the centralized setting.

More importantly, the approach described here offers a general framework for the design of algorithms for learning from distributed data that is provably exact with respect to its centralized counterpart. Central to this approach is a clear separation of concerns between hypothesis construction and extraction of sufficient statistics from data, making 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 specified set of constraints Z that describe the query capabilities and operations permitted by the data sources (e.g., execution of user supplied procedures).

FUTURE TRENDS

It is inevitable that autonomously developed data sources are semantically heterogeneous. The ontological commitments associated with a data source (and hence its implied semantics) are typically determined by the data source designers, based on their understanding of the intended use of the data. Very often, data sources that are created for use in one context or application find use in other contexts or applications, and therefore, semantic differences among autonomously

designed, owned, and operated data repositories are simply unavoidable. Effective use of multiple sources of data in a given context requires reconciliation of semantic differences. There have been significant community-wide efforts aimed at the construction of ontologies (e.g., Gene Ontology for molecular biology – GO at http://www.geneontology.org, Semantic Web for Earth and Environmental Terminology – SWEET at sweet.jpl.nasa.gov, etc.). Hence, there is an urgent need for methods that can dynamically and efficiently extract and integrate information needed for knowledge acquisition, from semantically heterogeneous data, from a user's perspective. Autonomous data sources often describe, or, because of privacy considerations, expose data at different levels of abstraction. The approach described here lends itself to adaptation to settings where the ontologies associated with the individual data sources differ from each other (Caragea et al., 2005) and to settings that require learning predictive models from partially specified data (Zhang et al., 2006).

CONCLUSION

With the proliferation of large, autonomous, distributed databases in many application domains, the problem of mining distributed data is an extremely important topic with relevance to many application domains where databases play a central role (e.g., bioinformatics, ecological informatics, e-commerce, health informatics). Effective solution of this problem is critical to advances in cyberinfrastructure for e-science (Hey & Trefethen, 2005). In this entry, we have precisely formulated the problem of learning classifiers from distributed data 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. The resulting algorithms are *provably exact* in that the hypothesis 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. The approach described can be extended to scenarios where the distributed data sources are semantically heterogeneous (Caragea et al., 2005).

Acknowledgements: This work is supported by the National Science Foundation Grant 0711396.

REFERENCES

Caragea, D., Silvescu, A., & Honavar, V. (2004). A Framework for Learning from Distributed Data Using Sufficient Statistics and its Application to Learning Decision Trees. International Journal of Hybrid Intelligent Systems. Vol 1. pp. 80-89.

Caragea, D., Zhang, J., Bao, J., Pathak, J., & Honavar, V. (2005). Algorithms and Software for Collaborative Discovery from Autonomous, Semantically Heterogeneous, Distributed, Information Sources. Invited paper. In: Proceedings of the Conference on Algorithmic Learning Theory. Lecture Notes in Computer Science. Vol. 3734. Berlin: Springer-Verlag. pp. 13-44.

Casella, G., & R.L. Berger, R.L. (2001). Statistical Inference. Duxbury Press,
Belmont, CA.

Domingos, P. (1997). Knowledge acquisition from examples via multiple models. In Proceedings of the Fourteenth International Conference on Machine Learning, pages 98–106, Nashville, TN. Morgan Kaufmann.

Duda, R., Hart, E., & Stork, D. (2000). Pattern Recognition. New York: Wiley.

Ozsu, M. T. & Valduriez, P. (1999). Principles of distributed database systems (2nd Edition). Prentice Hall, Inc.

Friedman, N., Getoor, L., Koller, D., & Pfeffer, A. (1999). Learning probabilistic relational models. In Proceedings of the Sixteenth International Joint Conference on Artificial Intel ligence, pp. 1300–1309, Orlando, FL, July 1999. Morgan Kaufmann Publishers Inc.

Mitchell, T. (1997). Machine Learning. New York: Addison-Wesley.

Srivastava, A., Han, E., Kumar, V. & Singh, V. (1999). Parallel formulations of decision-tree classification algorithms. Data Mining and Knowledge Discovery, 3(3):237– 261.

Grossman, L.R. & Gou, Y. (2001). Parallel methods for scaling data mining algorithms to large data sets. In J.M. Zytkow, editor, Handbook on Data Mining and Knowledge Discovery. Oxford University Press.

Hey, T., & Trefethen, A. E. (2005). Cyberinfrastructure for e-Science. Science, 308(5723), 817-821.

Park, B. & Kargupta, H. (2002). Distributed data mining: algorithms, systems, and applications. In Nong Ye, editor, Data Mining Handbook, pages 341–358. IEA.

Prodromidis, A. L., Chan, P. & Stolfo, S.J. (2000). Meta-learning in distributed data mining systems: issues and approaches. In H. Kargupta and P. Chan, editors, Advances of Distributed Data Mining. AAAI Press.

Provost, F.J. & Kolluri, V. (1999). A survey of methods for scaling up inductive algorithms. Data Mining and Knowledge Discovery, 3(2):131–169.

Zhang, J., Kang, D-K., Silvescu, A. & Honavar, V. (2006). Learning Compact and Accurate Naive Bayes Classifiers from Attribute Value Taxonomies and Data. In: *Knowledge and Information Systems*. Vol. 9. No. 2. pp. 157-179.

KEY TERMS AND THEIR DEFINITIONS

Machine Learning: A key objective of Machine Learning is to design and analyze algorithms that are able to improve the performance at some task through experience. A machine learning system is specified by several components: a) *Learner* - an algorithm or a computer program that is able to use the experience to improve its performance. b) *Task*: A description of the task that the learner is trying to accomplish (e.g., learn a concept, a function, etc.). c) *Experience*: Specification of the information that the learner uses to perform the learning. d) *Background*

knowledge: the information that the learner has about the task before the learning process (e.g. ”simple” answers are preferable over ”complex” answers). e) *Performance Criteria*: Measure the quality of the learning output in terms of accuracy, simplicity, efficiency etc.

Classification Task: A task for which the learner is given experience in the form of labeled examples and it is supposed to learn to classify new unlabeled examples. In a classification task, the output of the learning algorithm is called hypothesis or classifier (e.g., a decision tree, a support vector machine, etc.).

Learning from Data: Given a data set, a hypothesis class (e.g., decision trees), and a performance criterion (e.g., accuracy), a learning algorithm outputs a hypothesis that optimizes the performance criterion.

S**ufficient Statistics:** A statistic is called a sufficient statistic for a parameter if the statistic captures all the information about the parameter, contained in the data. More generally, a statistic is called a sufficient statistic for learning a hypothesis using a particular learning algorithm applied to a given data set, if there exists an algorithm that takes as input the statistic and outputs the desired hypothesis. A query that returns a statistic is called a statistical query.

Learning Task Decomposition: A learning algorithm can be decomposed in two components: (a) an *information extraction* component that formulates and sends a statistical query to a data source; and (b) a *hypothesis generation* component that uses the resulting statistic to modify a partially constructed algorithm output (and further invokes the information extraction component

if needed to generate the final algorithm output).

Distributed Data Sources: In a distributed setting, the data are distributed across several data sources. Each data source contains only a fragment of the data. This leads to a fragmentation of a data. Two common types of data fragmentation are: *horizontal fragmentation*, wherein (possibly overlapping) subsets of data tuples are stored at different sites; and *vertical fragmentation*, wherein (possibly overlapping) sub-tuples of data tuples are stored at different sites. More generally, the data may be fragmented into a set of relations (tables of a relational database, distributed across multiple sites).
stored at different sites.

Learning from Distributed Data: Given several fragments of a data set (distributed across several sites), a set of constraints (referring to privacy concerns, storage issues, operations allowed, etc.), a hypothesis class and a performance criterion, the task of the distributed learner is to output a hypothesis that optimizes the performance criterion, without violating the set of constraints. We say that an algorithm for learning from distributed data is *exact* if the hypothesis that it outputs is identical to that produced by the centralized algorithm for learning from the complete data set, obtained by appropriately combining the distributed data fragments.