Knowledge as a Service and Knowledge Breaching

Shouhuai Xu and Weining Zhang
Department of Computer Science, University of Texas at San Antonio
{shxu, wzhang}@cs.utsa.edu

Abstract

In this paper, we introduce and explore a new computing paradigm we call knowledge as a service*, in which a knowledge service provider, via its* knowledge server*, answers queries presented by some* knowledge consumers*. The knowledge server's answers are based on knowledge models that may be expensive or impossible to obtain for the knowledge consumers. While this new paradigm of computing is promising, we must establish a solid foundation to ensure its utility. We focus on the security aspect of the paradigm, and particularly on the problem we call* knowledge breaching *attack, which may allow an adversary to recover the knowledge underlying a knowledge service. Without being able to adequately handling such an attack, the knowledge service providers would never have any economic incentives to develop such a paradigm. Unfortunately, this paper theoretically shows that any* interesting knowledge *is subject to the knowledge breaching attack, and empirically shows that some knowledge models could be breached after a very small number of queries (e.g., 0.2-1% portion of the domain). Thus we need to investigate technical means that can alleviate such powerful attacks (at least for most practical knowledge models).*

1. Introduction

With the remarkable success of knowledge discovery and data mining techniques, we now envision a new service-oriented computing paradigm, called *knowledge as a service*, which offers new types of services based on knowledge typically extracted from large volumes of data owned and maintained by different parties. Our motivation is perhaps best described by the following example.

Example 1.1. Bob went to the GreatLife Insurance company to buy a life insurance. To determine his premium, the agent at GreatLife needs to know if Bob is likely to be involved in severe car accidents, so that a higher premium will be charged if Bob fits the profile of drivers who frequently cause severe car accidents. Since Bob has previous auto insurance from two companies, the EastInsurance and the WestInsurance, a predictive model extracted from these companies' databases will be a great help to the agent at GreatLife. However, due to concerns of client privacy and business competition, none of the two auto insurance companies would allow GreatLife or the other company to mine its data for such a knowledge model. Fortunately, GreatLife has subscribed a service from a knowledge dissemination company, KDC, which can extract a useful knowledge model from the data of the two auto insurance companies by using an advanced privacy-preserving data mining technology and use that model to answer, for a service fee, the GreatLife agent's question about Bob's likelihood of auto accident. Using the information obtained from KDC, the agent at GreatLife quickly determines a premium that simultaneously satisfied Bob and minimized the company's risk.

The service provided by KDC in this (not so) fictional example is what we call a knowledge service, where by *knowledge* we mean *knowledge models* such as decision trees, association rules, or neural networks[1].

While the paradigm of knowledge as a service is promising, it must be based on a solid foundation, which, in particular, must deal with important security issues so that knowledge service providers can get their investment paid off. Besides general security problems such as controlled access to the extracted knowledge and privacy protection in both knowledge extraction (i.e., to protect data privacy) and knowledge utilization (i.e., to protect privacy of queries of a knowledge consumer [13, 4]), this paradigm faces a new security issue called *knowledge breaching attacks* whereby an adversary may recover the knowledge underpinning a knowledge service. Without an adequate protection against this attack, knowledge as a service could hardly take off because service providers might fear not getting their investment paid off.

In this paper, we make the following contributions.

1 To precisely define the term *knowledge* is a challenge and is beyond the scope of this paper.

Proceedings of the 2005 IEEE International Conference on Services Computing (SCC'05)
0-7695-2408-7/05 $20.00 © 2005 IEEE

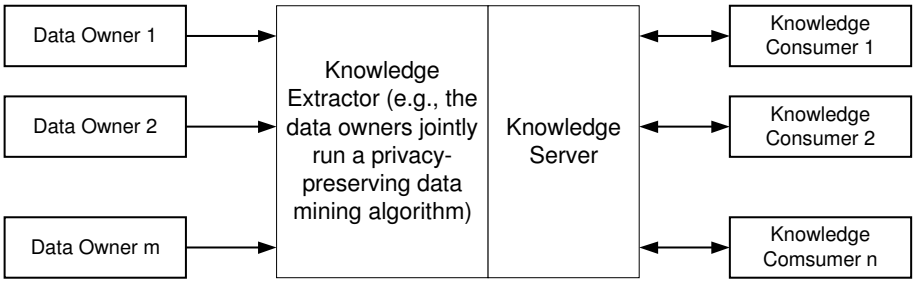

Figure 1. The paradigm of knowledge as a service

1. We introduce the knowledge as a service paradigm, and explore a framework to fulfill it.
2. We define the notion of *knowledge breaching* with respect to a large class of knowledge service that has a query-answer format and may be based on many different knowledge models (e.g., decision trees and association rules).[2] In particular, we theoretically show that any *interesting knowledge* (defined in the next section) is subject to knowledge breaching attacks. Furthermore, we empirically show that with two specific knowledge breaching strategies, some knowledge models could be breached after a small number of queries (e.g., 0.2-1% of the domain).
3. We observe that knowledge breaching is unique to the knowledge as a service paradigm, and cannot be resolved with traditional information security principles. Therefore, we suggest research directions for dealing with the powerful knowledge breaching attacks.

The rest of this paper is organized as follows. In Section 2, we present the framework of the knowledge as a service paradigm. In Section 3, we define the notion of knowledge breaching for a classification service. In Section 4, we describe two potential methods that an adversary may deploy. In Section 5, we present results from our systematic experimentation with the breaching methods against some knowledge models. In Section 6, we discuss some methods and research direction that deal with knowledge breaching. Related work is discussed in Section 7, and the conclusion is in Section 8.

2. Knowledge as a Service

Figure 1 depicts a high-level framework of the "knowledge as a service" paradigm, which involves three types of logical participants: *data owners*, *service providers*, and *knowledge consumers*. All participants, including the adversary, are probabilistic polynomial-time algorithms.

In general, knowledge consumers receive service based on knowledge that may be extracted by a service provider from one or more data owners' datasets. In this paper, we focus on knowledge services that have a question-answer format in an application context. With this type of service, a knowledge consumer sends the server a query of a specific format, and the server sends back an answer according to the underpinning knowledge model. More specifically,

- The data owners collect data from their daily business transactions. Although they are responsible for protecting the secrecy of the data, they are allowed to utilize the data, and the derivate thereof, in their own decision-making procedures.
- A knowledge service provider delivers knowledge service via its *knowledge server*, where the knowledge is extracted from the datasets through an appropriate knowledge extractor, such as a privacy-preserving data mining algorithm. We stress that the service provider, while owning the knowledge, does not necessarily own any data or learn any specific information about the data from which the knowledge is extracted[3].
- A knowledge consumer consults a knowledge service in its decision making procedures. Further, the knowledge consumer may also need to preserve the secrecy of its own customers' data, meaning that the query instance may not have to be exposed to the service provider.

2 We believe our definitions and methodologies can be naturally extended to deal with other classes of knowledge services, which may be less popular though.

3 We remark that while previous privacy-preserving data mining algorithms (e.g., [16]) focus on the case of the participants are all data owners, it is easy to extend the case to accommodate a new participant in a secure multi-party computation such that the new participant does not contribute any input but gets the output exclusively. We remark that this is even simpler in the approach of [1]

3. Knowledge Breaching Attacks

A *knowledge breaching attack* is a malicious activity against a knowledge server, which is launched by an adversary who plays the role of a knowledge consumer or a set of colluding knowledge consumers and aims to recover the target knowledge underpinning the knowledge service. The adversary attacks by making a *small* number of adaptively selected queries and using the answers to construct a learned knowledge, which in many application scenarios is only required to have a functionality sufficiently similar to that of the target knowledge.

3.1. Knowledge Functions and Knowledge Leak

It is instrumental to view both target and learned knowledge as functions that assign labels to instances in an application domain. This general viewpoint can easily accommodate services based on different knowledge models such as decision trees and association rules.

Let the domain $U = \{u_1, \ldots, u_N\}$ be a finite set of instances (or points) and the range C be a finite set of labels. Let $F = \{f \mid f : U \rightarrow C\}$ be a set of knowledge functions. From the adversary's point of view, the unknown *target function* is represented by a random variable f_T that has a probability distribution D_F over F such that $\Pr(f_T = f)$ is the probability that f is the target function. We assume that the adversary knows D_F, which is the worst-case scenario where the adversary can coordinate and adaptively choose all the queries. To learn the target function, the adversary queries the service for labels of a set of instances and uses these labels to derive a *learned function* f_L, which is also in F. Whether the adversary can effectively learn the target function with a small number of queries depends on how much information the labels of queried instances tell the adversary about the labels of remaining instances. To the adversary, each $u \in U$ is associated with a random variable X_u, which represents the label assigned to u by the unknown target function, that is, $f_T(u)$, and has a probability distribution D_u such that $\Pr(X_u = c)$ is the probability that $f_T(u) = c$ and $\Pr(X_{u_1} = c_1, \ldots, X_{u_k} = c_k)$ is the joint probability that $f_T(u_i) = c_i$, where $i = 1, \ldots, k$ and $k \leq N$.

Definition 3.1. Let $Q \subset U$ be a set of points asked by the adversary (or query points), $S = \{< u, c > \mid u \in Q, c = f_T(u)\}$ be the query-answer pairs, and $\Pr(X_u = c \mid S)$ be the probability that $f_T(u) = c$ given S. An point $u \in U - Q$ is *sensitive* to Q if $\exists c \in C$, such that, $\Pr(X_u = c \mid S) \neq \Pr(X_u = c)$. The target function has a *knowledge leak* if $\exists Q \subset U$ and $\exists u \in U - Q$ such that u is sensitive to Q.

Intuitively, a knowledge leak gives a hint to the adversary about what labels the target function may assign to the not-yet-queried (or unseen) points. We say a knowledge (function) is *interesting* if it describes certain hidden patterns of the data, therefore, assigns labels to points dependently; *non-interesting* otherwise. The following theorem shows that any interesting knowledge is vulnerable to a knowledge breaching. (The proof is omitted due to the lack of space. See [25])

Theorem 3.1. If $\Pr(f_T = f) \neq \Pi_{u \in U} \Pr(X_u = f(u))$, then f has a knowledge leak.

Notice that both Definition 3.1 and Theorem 3.1 address knowledge leak qualitatively and provide no quantitative measure of such a leakage. To devise an ideal metric of knowledge breaching, we need to take into consideration the adversary's a priori knowledge, which is unfortunately not available to us. Thus in the rest of this paper, we simulate the adversary by using the probability that the adversary's learned function correctly labels an unseen point as a simple measure for knowledge breaching. Formally,

Definition 3.2. Let $Q \subset U$ be a set of queried points, $p_{f_L,Q}$ be the probability that the learned function f_L correctly labels a point in $U - Q$, and $0 \leq p_0 \leq 1$ be a real number. We say that f_L causes a *degree* p_0 *knowledge breaching* of f_T if $p_{f_L,Q} > p_0$.

Typically, $p_{f_L,Q}$ is not directly available to the adversary. However, the adversary may estimate it statistically by making some queries, as shown in the next section.

Algorithm KFB:
Input: Service S, domain U of target function f_T, accuracy threshold p_0, significance level α
Output: Learned function f_L after querying Q such that $\Pr(p_{f_L,Q} > p_0) = 1 - \alpha$
Method:

1. DeriveInitialFunction(f_L, U, S);
2. while TestAccuracy(S, f_L, U, p_0, α) is not successful do
3. Refine(f_L, S);
4. return f_L.

Figure 2. A General Algorithm for Knowledge Function Breaching

4. Two Methods of Knowledge Breaching

In this section, we describe two knowledge breaching methods, region split and active learning, which are specialization of the general algorithm shown in Figure 2. In this algorithm, an initial syntactic representation of the learned function is first constructed (step 1), and then repeatedly

tested (step 2) and refined (step 3) until the resulting learned function satisfies a given accuracy threshold (p_0) with a required confidence ($1 - \alpha$). Each of these steps needs to query the service for one or more points. The two methods differ in steps 1 and 3, and in the representation of the learned functions. However, step 2 is identical in both methods, where a statistical hypothesis test is used to check if the learned function satisfies the accuracy thresholds. In the following we give a rather brief presentation of the methods due to the lack of space. More details can be found in a longer version of this paper.

4.1. Region Split Method

In the region split method, the learned function is represented as a list of cube-shaped regions of an n-dimensional domain U, which is the result of recursive split of the domain. Each region is represented by its corner points and their labels which are obtained through queries. To find the label of a point inside a region, a simple majority votes among the region's corner points is taken and the winning label is assigned to the point. If there is a tie of two or more labels, one label is random chosen to break the tie. A region needs to be split if it contains a known mislabeled point. Thus, a region split refines the learned function.

4.1.1. Region Split To ease the presentation, consider a domain U where each dimension A_i is a finite interval $[L_i, H_i]$ of integers, where $1 \leq i \leq n$. A region in U is defined by $r = \{< v_1, \ldots, v_n > |\ v_i \in [l_i, h_i], L_i \leq l_i \leq h_i \leq H_i, 1 \leq i \leq n\}$. The set of corner points of r is defined by $\{< v_1, \ldots, v_n > |\ v_i = l_i \vee v_i = h_i\}$. By this definition, a region can not be empty.

To split the region, for each dimension A_i such that $l_i < h_i$, the interval $[l_i, h_i]$ is split as evenly as possible into two sub-intervals $[l_i, ml_i]$ and $[mh_i, h_i]$, where $ml_i = \lfloor (h_i + l_i)/2 \rfloor$ and $mh_i = ml_i + 1$. Now a sub-region of r is the set of points $\{< v_1, \ldots, v_n > |\ v_i \in I_i\}$, where interval $I_i = [l_i, ml_i]$ or $[mh_i, h_i]$. Notice that these sub-regions are pair-wise disjoint and that a single-point region cannot be split any further. After a new region is obtained, its corner points are labeled by querying the service.

4.1.2. Test of the Accuracy Hypothesis The accuracy of f_L is tested by a standard statistical hypothesis test [22] that check if $\Pr(p_{f_L,Q} > p_0) > 1 - \alpha$ using a random sample of size m. Here, $p = p_{f_L,Q}$ is the probability that a point is correctly labeled by f_L. Let $H_0 : p = p_0$ and $H_a : p > p_0$ be the null and the alternative hypothesis, respectively. The sample probability $\hat{p}$ is used as an estimator in the Z-test to see if H_0 can be rejected (that is, $Z > z_\alpha$, where

$$Z = \frac{\hat{p} - p_0}{\sqrt{\frac{p_0(1-p_0)}{m}}}$$

and z_α is the value satisfying $P(Z > z_\alpha) = \alpha$.) The sample size can be determined based on [22]. In our experiments, a 5% more points is sampled to ensure that the test-refinement loop would terminate.

4.1.3. Refine the Learned Function If the learned function fails the accuracy test, it must contains some problematic region that contains at least one mislabeled point. To refine, these problematic regions are further split. One way to refine is to make one split for each problematic region. An alternative is to split each problematic region repeatedly until every sub-region has consistently labeled corner points (that is, they have the same labels). While the first option may do too little in each round and therefore cause too many queries (needed by many rounds of test and refine steps), the second option may do too much in one round even when the accuracy threshold is not that high therefore waste queries (needed to label new regions' corner points). To solve these problems, we adopt a heuristic method that performs repeated split similar to the second option but interleaves the splits with pseudo hypothesis tests that uses the sample taken in step 2 instead of fresh ones. This heuristic aims to save the number of queries while it still allows regions to be split deeply enough to avoid mislabeling.

4.2. Active Mining Method

We consider a well known active learning algorithm [15], which trains, on the same set of training instances, two classifiers: a probabilistic classifier, which is used to select training instances, and a decision tree classifier. The training instances are selected in passes until their number reaches a given threshold. Initially, the probabilistic classifier is trained on three instances given by an expert. In each subsequent pass, the probabilistic classifier is used to estimate a label and a certainty factor for each unlabeled instance. Then, four instances whose estimated labels are the least certain are selected, labeled by the expert, and combined with existing training instances. The probabilistic classifier is then retrained on the expanded set of training instances. This algorithm cannot be directly used in our setting for following reasons.

1. Its termination condition is based on a threshold of the number of training samples rather than the accuracy of the learned model.
2. The number of newly selected training instances (namely, four) is too small compared to the number of random samples needed by our accuracy hypothesis test.
3. It relies on human experts for initial training samples.
4. Its first classifier is limited to a binary classification.

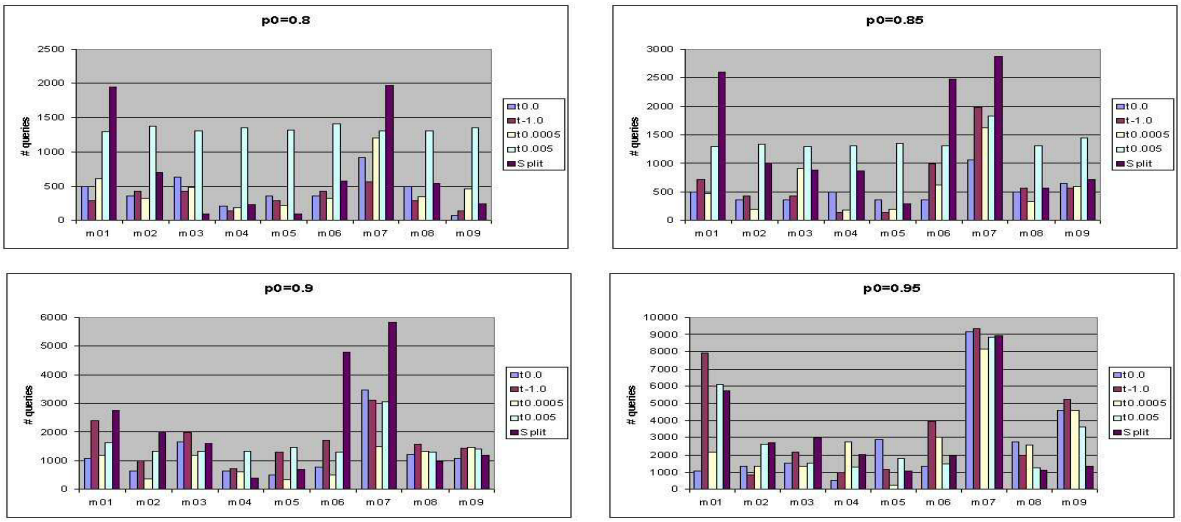

Figure 3. Number of Queries

To adapt the active learning algorithm for our purpose, we extend it in following ways. First, a standard hypothesis test is used as the termination condition. Second, in each pass, two sets of new training samples are added to the existing training set: the set of samples obtained via the uncertainty sampling and the set of samples used in the hypothesis test. Third, instead of using a fixed small number of initial training samples provided by a human expert, we allow the algorithm to choose some random points as the initial training samples, and query the service to obtain their labels. Finally, in order to accommodate multiple labels, we require that the first classifier is able to estimate a probability distribution of labels for any given point in the function domain, and uses this probability distribution to compute the certainty of the estimated labels. Due to limited space, the details are omitted. The second classifier trained by the modified active learning method is the learned function.

5. Experiments

We implemented the two knowledge breaching methods described in Section 4 and applied them to nine manually created target functions (referred to as m01, m02, . . ., m09), which are binary functions over a 2-dimensional domain of 500×500 points and have a variety of complexities. In our experiments, we vary the accuracy threshold p_0 from 0.8 to 0.95 with an increment of 0.05, and fixed the significance level at $\alpha = 0.05$. For active learning method, we scaled the size of the initial training samples from 0 to 20% of the domain size.

We measured the performance by the *number of queries* including those for creating the initial learned function, performing the accuracy hypothesis tests, and refining the learned functions, and the *number of refinements* needed to derive learned functions. The size of random samples used in the hypothesis test is 71.

We compare performances of the region split method (referred as split) and four different setups of the active learning method according to the size of the initial training set: (1) with no initial sample (referred as t0.0), (2) with initial set of the size of the random samples used in the hypothesis test, in this case 71 (referred as t1.0), (3) with a set containing 0.05% points of the domain (referred as t0.0005), and (4) with a set containing 0.5% points of the domain (referred as t0.005). Below we simply refer to these testing conditions by the names in parenthesis, and treat them as if they are different algorithms.

5.1. Results and Discussion

In Figures 3 and 5, each of the four charts corresponds to a different value of accuracy threshold p_0, in which the horizontal axis identifies target functions and the vertical axis is the number of queries (in Fig. 3) or refinements (in Fig. 5). For each target function, the charts show the performance of each of the above-mentioned five algorithms, as identified by the legend.

5.1.1. Function Breaching is Inevitable Fig. 3 clearly shows that all the five algorithms can successfully derive the learned functions with respect to each given accuracy threshold. However, the performance seems to reflect to some extent the inherent complexity of target functions. For example, the most complex function m7 has much worse performance than the simplest function m4.

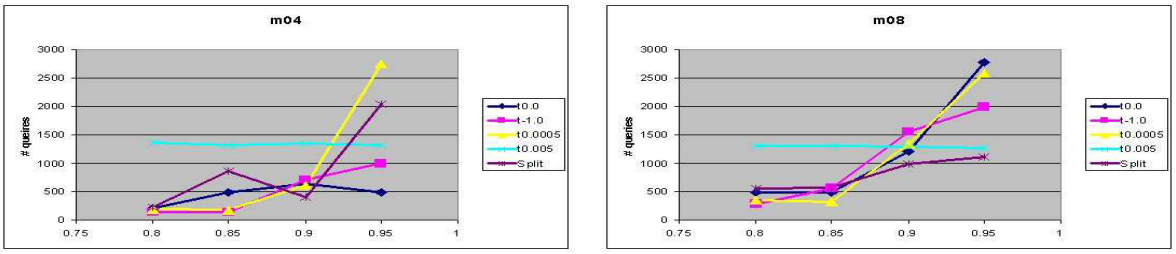

Figure 4. Effect of Accuracy Threshold

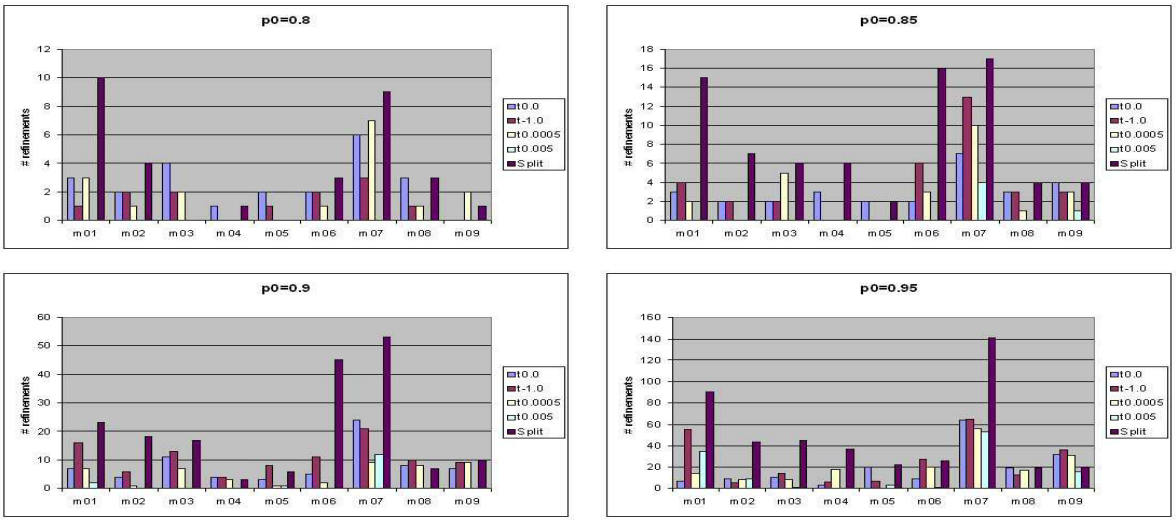

Figure 5. Number of Refinements

5.1.2. Effect of Accuracy Threshold Fig. 4 shows the number of queries of two target functions under the five algorithms, where the horizontal axis is the accuracy threshold, and the curves represents the performance of the algorithms. Although most tests confirm the intuition that the higher the accuracy threshold is the more the queries are required, exceptions do exist. For example, with target function m04, algorithm t0.0 issues less queries for $p_0 = 0.95$ than for $p_0 = 0.9$. Similarly algorithm split issues less queries for $p_0 = 0.9$ than for $p_0 = 0.85$. A preliminary interpretation is that while for t0.0, the exception may be caused by the fact that the size of the initial training set is 0, for split, the exception is perhaps due to the characteristics of m04.

5.1.3. Effect of the Number of Initial Training Samples The number of initial training samples is an important parameter in the active learning method. For a given accuracy threshold, it holds that (1) the more the training samples are used in the Step 1, the less the number of refinement is needed; and (2) when the number of initial training samples reaches a certain threshold, no refinement is needed (see the curve corresponding to algorithm t0.005 in Fig. 5) and thus the active learning method actually degenerates to a traditional passive learning method.

On the other hand, when the number of initial training samples is less than a certain threshold, the dependency of the number of queries on the number of initial training samples is elusive. For example, as shown in Fig. 3 none of split, t0.0, t-1.0, and t0.0005 always outperforms the others.

6. Discussion

In the last two sections we theoretically and empirically showed that knowledge breaching is a powerful attack against some *interesting* knowledge functions or models. A natural next question is: Can we efficiently deal with such attacks? It would be ideal if we can completely block such attacks. However, despite the existing solutions to the various security problems, we observe that traditional security principles cannot adequately deal with this attack. This is so because an adversary would have to be allowed to query

a knowledge server with respect to instances chosen by the adversary. For example, the above results have showed that a notion similar to the "security against an adaptive adversary" in cryptography is doomed.

One approach to mitigating the knowledge breaching attack would be to limit the number of queries a knowledge consumer is allowed to present. (This strategy has been deployed in statistical database systems to limit the information that can be inferred by an adversary; see Section 7 for further discussion.) However, this approach is very restrictive because, after all, the primary motivation of knowledge as a service is to maximize the revenue of the knowledge service provider while preserving the privacy of various legitimate participants and satisfying the need of the knowledge consumers.

A better approach is to allow the knowledge server to charge the knowledge consumers according to an appropriate price mechanism. However, the problem of determining such a price mechanism is challenging because it should be based on the *average*- or even the *best*-case learning complexity bounds from the adversary's perspective. We stress that the learning complexity bounds in computational learning theory are typically in the *worst*-case scenarios. It is not clear how we can derive the *average-/best*-case learning complexity bounds, given that there are possibly infinitely many ways to breach a knowledge function. As a first step towards this ultimate goal, we have found a heuristic method whereby an adaptive pricing mechanism can be imposed. We will fully explore it in a future work.

7. Related Work

On database as a service and knowledge sharing. Service-oriented computing is an active research area and a number of service types can be identified, including application as a service, database as a service [14], data mining model as a service [19], and the general notion of web service. The knowledge as a service presented in this paper is a natural evolution of this trend and helps bridge the deviation of knowledge utilization from knowledge extraction.

The sharing of data mining models has recently attracted increasingly more attention in the data mining community [9, 19, 23]. The Predictive Modelling Markup Language (PMML) [9] has been proposed by the Data Mining Group as a standard format of data mining models. A number of commercial as well as research prototype data mining systems, such as [23], are able to import/export PMML-based models. These techniques can be seamlessly integrated into knowledge as a service paradigm and are complimentary to the security aspect of knowledge services.

On privacy protection of data. Many useful techniques for privacy protection have been contributed by the cryptography community [6, 7, 8]. These techniques protect users' anonymity while allowing them to show their legitimacy. On the other hand, inference control in statistical databases, which allows accesses to statistics about groups of entities while protecting the confidentiality of the individual entities, have been extensively investigated [10, 18, 24]. We remark that all these techniques don't address the problem of knowledge breaching in the context of knowledge as a service.

On privacy-preserving data mining and computational machine learning. It is interesting that on the one hand, the paradigm of knowledge as a service accommodates (privacy-preserving) data mining as a useful component, and on the other hand, knowledge can be compromised via data mining and machine learning algorithms.

There are two approaches to privacy-preserving data mining. The first approach is to randomize the values in individual records [1]. A model is then built over the randomized data, after first compensating for the randomization (at an aggregate level). This approach is potentially vulnerable to privacy breaches: based on the distribution of the data, one may be able to learn with high confidence that some of the randomized records satisfy a specified property (even though privacy is preserved on average). In general, this approach is still in its early stage (see [11] for the subtleties in capturing the right notion of privacy) and does not provide accurate knowledge. The second approach is based on cryptographic secure multi-party computation techniques [26, 13, 16]. This approach does provide accurate knowledge and a strict privacy guarantee, but is typically much less efficient. In spite of some recent advances in cryptography (e.g., [12]), significant performance improvements are very much needed.

As a theoretical branch of machine learning, the computational learning theory has been focused on identifying concept classes that can be learned by polynomial algorithms with polynomial number of training samples in various theoretical learning models [21, 2, 17, 3]. Our result in Theorem 3.1 is in an information-theoretic sense because there is no assumption on the adversary's computational capability. Thus, one may wonder if a probabilistic polynomial-time adversary can efficiently learn the target knowledge with a small number of queries. This is related to the computational learning theory [20], which is primarily concerned with the complexity of learning in theoretical models such as learnability results and complexity bounds in the *worst* case. However, the results in learning theory often do not provide practical solutions (or algorithms) because even if a knowledge function is not learnable in theory, it could still be significantly breached in practice. Further, what seems to be needed for our purpose is the *average* case or even *best* case complexity bounds .

Knowledge as a service vs. priced information. A no-

tion that is remotely related to the knowledge as a service paradigm is the so-called priced information [5].

8. Conclusion

We introduced a new computing paradigm called knowledge as a service. We explored a high-level framework fulfilling this paradigm, and focused on the crucial security issue of knowledge breaching attacks. We theoretically showed that any knowledge service based on interesting knowledge function is vulnerable to these attacks. Through systematic experiments (with various heuristic optimizations), we also empirically showed that some knowledge models can be breached quickly. Finally, we discussed why traditional security principles cannot solve the problem of knowledge breaching, and suggested research directions towards dealing with these powerful attacks.

Acknowledgement

The authors wish to thank an anonymous reviewer for many constructive critics and suggestions, which lead to improvement of this paper's quality.

References

[1] R. Agrawal and R. Srikant. Privacy-preserving data mining. In *ACM SIGMOD International Conference on Management of Data*, pages 439–450. ACM, 2000.

[2] D. Angluin. Queries and concept learning. *Machine Learning*, 2(4):319–342, Apr. 1988.

[3] D. Angluin and P. Laird. Learning from noisy examples. *Machine Learning*, 2(4):343–370, 1988.

[4] D. Beaver, J. Feigenbaum, and V. Shoup. Hiding instances in zero-knowledge proof systems (extended abstract). In *10th Annual International Cryptology Conference (CRYPTO'90)*, pages 326–338, 1990.

[5] M. Charikar, R. Fagin, V. Guruswami, J. Kleinberg, P. Raghavan, and A. Sahai. Query strategies for priced information. In *ACM Symposium on Theoretical Computer Science*, pages 484–493. ACM, 2000.

[6] D. Chaum. Untraceable electronic mail, return addresses, and digital pseudonyms. *Commun. ACM*, 24:84–88, Feb. 1981.

[7] D. Chaum. Blind signatures for untraceable payments. In R. L. Rivest, A. Sherman, and D. Chaum, editors, *Advances in Cryptology – CRYPTO 1982*, pages 199–203, New York, 1983. Plenum Press.

[8] D. Chaum. Security without identification: Transaction systems to make big brother obsolete. *Communications of the ACM*, 28(10):1030–1044, Oct. 1985.

[9] Data Mining Group. PMML version 2.1. http://www.dmg.org, March 2003.

[10] D. Denning. *Cryptography and Data Security*. Addison-Wesley, Mass., 1982.

[11] A. Evfimievski, J. Gehrke, and R. Srikant. Limiting privacy breaching in privacy preserving data mining. In *ACM Symposium on Principles of Database Systems*, pages 211–222. ACM, 2003.

[12] M. Freedman, K. Nissim, and B. Pinkas. Efficient private matching and set intersection. In C. Cachin and J. Camenisch, editors, *Advances in Cryptology - EUROCRYPT 2004*, volume 3027 of *Lecture Notes in Computer Science*, pages 1–19. Springer, 2004.

[13] O. Goldreich, S. Micali, and A. Wigderson. How to play any mental game or a completeness theorem for protocols with honest majority. In *Proc. 19th ACM Symp. on Theory of Computing*, pages 218–229. ACM, 1987.

[14] H. Hacigümüs, S. Mehrotra, and B. Iyer. Providing database as a service. In *IEEE International Conference on Data Engineering*, pages 29–38. IEEE Computer Society, 2002.

[15] D. D. Lewis and J. Catlett. Heterogeneous uncertainty sampling for supervised learning. In *Proceedings of the Eleventh International Conference on Machine Learning*, pages 148–156, 1994.

[16] Y. Lindell and B. Pinkas. Privacy preserving data mining. In M. Bellare, editor, *Advances in Cryptology – Crypto 2000*, pages 36–54. Springer, 2000. Lecture Notes in Computer Science No. 1880.

[17] N. Littlestone. Learning quickly when irrelevant attributes abound: A new linear-threshold algorithm. *Machine Learning*, 2(4):285–318, 1988.

[18] T. Lunt. Current issues in statistical database security. In C. Landwehr and S. Jajodia, editors, *Database Security V: Status and Prospects*, pages 381–385. IFIP WG 11.3, 1991.

[19] S. Sarawagi and S. H. Nagaralu. Data mining models as services on the Internet. *ACM SIGKDD Explorations*, 2(1):24–28, 2000.

[20] L. G. Valiant. A theory of the learnable. *Commun. ACM*, 27(11):1134–1142, Nov. 1984.

[21] L. G. Valiant. Learning disjunctions of conjunctions. In *Proc. of the 9th International Joint Conference on Artificial Intelligence, vol. 1*, pages 560–566, Los Angeles, California, 1985. International Joint Committee for Artificial Intelligence.

[22] D. D. Wackerly, W. M. III, and R. L. Scheaffer. *Mathematical Statistics with Applications*. Duxbury, sixth edition edition, 2002.

[23] D. Wettschereck and S. Muller. Exchanging data mining models with the predictive modelling markup language. In *International Workshop on Integration and Collaboration Aspects of Data Mining, Decision Support and Meta-Learning*, 2001.

[24] W. E. Winkler. Masking and re-identification methods for public-use microdata: Overview and research problems. In *Privacy in Statistical Databases*. Springer: New York, 2004.

[25] S. Xu and W. Zhang. Knowledge as service and knowledge breaching (full version). Technical report, Department of Computer Science, University of Texas at San Antonio, 2005.

[26] A. C. Yao. How to generate and exchange secrets. In *IEEE Symposium on Foundations of Computer Science*, pages 162–167, Toronto, 1986. IEEE.