The VLDB Journal (2006) 15:388–402
DOI 10.1007/s00778-006-0011-4

SPECIAL ISSUE PAPER

Privacy leakage in multi-relational databases: a semi-supervised learning perspective

Hui Xiong · Michael Steinbach · Vipin Kumar

Received: 30 September 2005 / Accepted: 25 May 2006 / Published online: 1 August 2006

Abstract In multi-relational databases, a view, which is a context- and content-dependent subset of one or more tables (or other views), is often used to preserve privacy by hiding sensitive information. However, recent developments in data mining present a new challenge for database security even when traditional database security techniques, such as database access control, are employed. This paper presents a data mining framework using semi-supervised learning that demonstrates the potential for privacy leakage in multi-relational databases. Many different types of semi-supervised learning techniques, such as the K-nearest neighbor (KNN) method, can be used to demonstrate privacy leakage. However, we also introduce a new approach to semi-supervised learning, hyperclique pattern-based semi-supervised learning (HPSL), which differs from traditional semi-supervised learning approaches in that it considers the similarity among groups of objects instead of only pairs of objects. Our experimental results show that both the KNN and HPSL methods have the ability to compromise database security, although the HPSL is better at this privacy violation (has higher prediction accuracy) than the KNN method. Finally, we provide a principle for avoiding privacy leakage in multi-relational databases via semi-supervised learning and illustrate this principle with a simple preventive technique whose effectiveness is demonstrated by experiments.

A preliminary version of this work has been published as a two-page short paper in ACM CIKM 2005 (Proceedings of the ACM conference on information and knowledge management (CIKM) 2005).

H. Xiong (✉)
MSIS Department, Rutgers University,
180 University Avenue, Newark, NJ 07102, USA
e-mail: hui@rbs.rutgers.edu

M. Steinbach · V. Kumar
Department of Computer Science and Engineering,
University of Minnesota, 4-192 EE/CS Building,
200 Union Street SE, Minneapolis, MN 55455, USA
e-mail: steinbac@cs.umn.edu

V. Kumar
e-mail: kumar@cs.umn.edu

1 Introduction

In multi-relational databases, a view, which is a context- and content-dependent subset of one or more tables (or other views), is often used to preserve privacy by hiding sensitive information. For instance, a view might be created to allow a sales manager to see only that portion of a customer table that is related to customers in the manager's own territory. This view might also be limited to selected columns from the base tables in which the subset of customer information is contained. Since a key purpose of database views is to hide sensitive information by controlling data access, the results of security breaches in database views can be serious, ranging from financial exposure to disrupted operations.

A concern for view security was initially raised by Codd's fundamental work on relational databases [8]. Indeed, one of the main issues faced by database security professionals is avoiding or limiting the inference capabilities of database users. Specifically, the goal is to limit the ability of users to employ information available at one security level to infer facts that should be protected at a higher security level.

In this paper, our concern with database security is different from the inference problem mentioned above.

More specifically, if some information from a higher security level is known by a user at a lower security level without authorization for some reason,[1] then this user may be able to predict additional information that should be protected at a higher security level. Indeed, the focus of this paper is to present a framework based on semi-supervised learning that illustrates the potential for this type of privacy leakage in database views.

Semi-supervised learning techniques attempt to apply both labeled and unlabeled data for predicting class labels for unlabeled objects. Among such techniques, there is a promising family of methods which are analogous to the traditional K-nearest-neighbor (KNN) method used in supervised learning [28]. The hypothesis behind these methods is that similar data objects tend to have similar class labels.

More recently, we have defined a new pattern for association analysis – the *hyperclique pattern* [33] – that demonstrates a particularly strong connection between the overall similarity of a set of attributes (or objects) and the itemset (local pattern) in which they are involved. The hyperclique pattern, described in more detail later, possesses the *strong affinity property*, i.e., the attributes (objects) in a hyperclique pattern have a guaranteed level of global pairwise similarity to one another as measured by the cosine similarity measure, which is also known as the uncentered Pearson correlation coefficient.[2] Intuitively, a hyperclique pattern includes objects that tend to be from the same class category. Based on this observation, we propose a new semi-supervised learning approach, the hyperclique pattern-based semi-supervised learning (HPSL) method. By considering the similarity among all objects in a hyperclique instead of the similarity between only pairs of objects, we can improve semi-supervised learning results more than those based on KNN approaches.

The main contributions of this paper can be summarized as follows.

- We show a new challenge for database security from the data mining/machine learning perspective. More specifically, we show that classic database security techniques may be inadequate in light of developments in semi-supervised learning. To demonstrate this, we present a framework that illustrates the potential for privacy leakage in database views with respect to semi-supervised learning.
- We introduce a new semi-supervised learning approach, the HPSL method. We use this technique, along with a couple of KNN semi-supervised learning approaches, to illustrate privacy leakage in database views. However, this framework is valid for all other semi-supervised learning techniques.
- We conduct extensive experiments on several real data sets to show the effectiveness of the HPSL method. Our experimental results show that the HPSL approach can achieve better prediction accuracy than traditional KNN techniques.

Overview The remaining paper is organized as follows. Section 2 formalizes the problem of information inference using semi-supervised learning, while in Sect. 3, we consider related work. We describe the basic concepts of hyperclique patterns in Sect. 4, present two KNN semi-supervised learning methods in Sect. 5, and introduce our HPSL method in Sect. 6. Section 7 describes an approach to protecting databases against the privacy leakage threat we have described. All experimental results, including those related to privacy protection, are given in Sect. 8. Finally, Sect. 9 draws conclusions and suggests future work.

Note that, in this paper, we use transaction data sets with binary variables. However, the issues and techniques described are also applicable to more general types of data. In particular, the hyperclique pattern, which was originally defined for binary transaction data can be applied to continuous data, either by transforming that data into a binary transaction format or by using the techniques for finding hyperclique patterns in continuous data that we recently proposed [30].

2 Problem formulation

A major purpose of this paper is to investigate privacy leakage in database views. We formalize the problem definition for privacy leakage in multi-relational databases via semi-supervised learning as follows:

Problem: privacy leakage in multi-relational databases given:

- ϑ is a database view that contains selected attributes, $I = \{i_1, i_2, \ldots, i_m\}$, potentially from several base tables.
- $C = \{c_1, c_2, \ldots, c_p\}$ is a set of attributes that are to be protected and are not provided in ϑ.
- $O = \{o_1, o_2, \ldots, o_n\}$ is a set of tuples (objects) in ϑ for which the values of attributes in C are unknown by policy.

[1] Information from a higher security level could be obtained in a variety of ways, for example, by eavesdropping – electronic or otherwise – but the details are outside the scope of this paper.

[2] When computing cosine similarity (the uncentered Pearson correlation coefficient), the data mean *is not* subtracted.

– $K = \{k_1, k_2, \ldots, k_l\}$ is a set of tuples (objects) in ϑ for which the values of attributes in C are known without authorization.

Approach:

– Use a semi-supervised learning mechanism to predict the values of attributes in the set C for objects in the set O.

Objective:

1. Predict, with high accuracy, the values of attributes in the set C for *some* objects in the set O.

Constraints

– $l \ll n$, where l is the number of objects whose class labels are known without authorization for some reason and n is the total number of objects.
– The attributes in the set I have predictive power for the attributes in the set C.

This problem is challenging, since the number of training objects is much smaller than the number of objects that are to be predicted. This is typically referred to as the small training sample size problem [13,27] in machine learning. Indeed, supervised classification techniques cannot obtain reliable results if only a very small set of samples are available. To this end, semi-supervised learning techniques [28], which make use of both unlabeled and labeled data, have recently been proposed to cope with the above challenge.

Example 1 Figure 1 gives an illustration of the problem. In the figure, the view contains m attributes and n tuples (objects). All the information in the view is known by the user. Also, there are p attributes that are in base tables but not in the view. Hence, the information in these p attributes is unknown to a user of the database view. However, if for some objects, these p attributes are known to a user of the database view, then such a user may use semi-supervised learning techniques to predict these p attributes for other objects. This is the problem addressed in this paper.

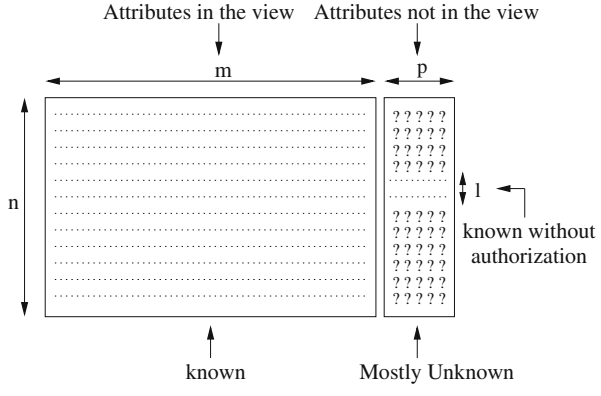


Fig. 1 Illustration of the problem

To make the problem concrete, let us introduce a real-world scenario as follows. A major advertising company, ABC Advertising, keeps a database of its several thousand customers. Much of this information is routine: names, addresses, visits by company representatives, etc. This information is not generally public, but often needs to be accessed by the administrative staff for mailings, to arrange visits, and for handling a wide variety of administrative chores. However, some information, such as the likelihood of the customer to defect to another ad agency, is sensitive and only available to a small number of employees. A low-paid administrative assistant at ABC has been offered a bribe by a competitor, XYZ Advertising, for any information that leads to XYZ winning a new account. With the assistant's help, XYZ is able to obtain a copy of the database information that is accessible to the assistant. The assistant is also able to provide a memo that lists 30 customers that were lost to other agencies last year. Using this information, the president of XYZ asks a data mining consultant to build a predictive model for the data. This model reveals that customers who are at risk of defecting get special mailings, as well as visits from both an account representative and a vice-president. Using this fact, XYZ is able to identify promising leads both from the database information that they stole and inside information that the administrative assistant continues to provide.

3 Related work

Related literature can be grouped into three categories: database security, semi-supervised learning, and privacy preserving data mining.

3.1 Database security

As mentioned, a goal of database security [17] is to limit the ability of users to infer facts that should be protected at a higher security level. Consider the following example.

Example 2 Given a multi-relational database, assume that there is a view named 'cargo' that contains information on the various cargo holds available on each outbound airplane. Each row in this view represents a

single shipment and lists the contents of that shipment and the flight identification number. The flight identification number may be cross-referenced with other base tables to determine the origin, destination, flight time, and other information. The 'cargo' view is presented in Table 1.

If a database user with a top secret security clearance requests information on the cargo carried by flight 1200, then this user would see all shipments. However, if a user without a security clearance requests the data, then this user would not see the top secret shipment. The above correctly implements the security rules that prohibit someone with lower security levels from seeing information with higher security levels. However, assume that there is a uniqueness constraint on flight ID and cargo (to prevent the scheduling of two shipments for the same hold). When a user without top secret clearance sees that nothing is scheduled for cargo hold C on flight 1200, this user might attempt to insert a new record to transport some vegetables on that flight. But the insertion of this record will fail due to the uniqueness constraint. At this point, the user can infer that there is a secret shipment on flight 1200 and could then cross-reference the flight information table to find out the source and destination of the secret shipment, as well as various other information.

There are two basic approaches to avoid such inference. One is polyinstantiation [10], which allows the creation of multiple instances of data records in a way such that a user with lower security levels can see the tuple associated with a particular primary key populated with one set of element values. However, a user with higher security levels may see the 'same' tuple with perhaps different values for some of the elements or multiple tuples of different levels with different data values. Another approach is to perform data access control and ensure that the set of all classification constraints is consistent [9]. In other words, even if users at lower security levels can know about the existence of a classified shipment, they will not have access to information about the contents of that shipment.

Table 1 The 'cargo' view

Flight ID	Cargo	Contents	Classification
1200	A	Boots	Unclassified
1200	B	Toys	Unclassified
1200	C	Guns	Top Secret
1200	D	Butter	Unclassified

In this paper, we raise another inference concern from a data mining perspective. Our hypothesis is that, if some information at a higher security level is known to a user at a lower security level, then it is possible that this user may infer more information at a higher security level. This problem can be treated as one of semi-supervised learning on labeled data (some leaked information at a higher security level) and unlabeled data (information at a lower security level) [28].

3.2 Semi-supervised learning

Traditional supervised learning methods build a prediction model from labeled data and use this model for predicting the labels of objects with unknown labels. However, in the real world, there are many situations where only a small fraction of the objects are labeled. In this case, many difficulties may arise in building a prediction model solely based on labeled data. This is referred to as the small training sample size problem [13,27] in machine learning.

Semi-supervised learning techniques [7,18,28,25], which make use of both unlabeled and labeled data, have recently been proposed to cope with the above challenge. Among such techniques, there is a promising family of methods that are similar to the KNN technique used in traditional supervised learning [28]. The hypothesis behind these methods is that similar data objects tend to have similar class labels. Our HPSL is also based on this hypothesis, but differs from the KNN approaches in that it considers the similarity among a group of objects instead of just the similarity between pairs of objects.

Finally, we should point out that the objective of most traditional semi-supervised learning methods is to learn class labels for all the objects, while our HPSL method only predicts the class labels for objects close to objects with known class labels.

3.3 Privacy preserving data mining

Privacy preserving data mining [1,3–6,12,14,16,21,22] strives to provide valid data mining results without revealing sensitive data values, i.e., without violating privacy. A traditional approach for privacy preserving data mining is to add random noise to the data in such a way that the individual data values are distorted while still allowing reconstruction of the original distributions of the values of the confidential attributes [3]. Note that the goal of this approach is to reconstruct distributions, not individual data values. Using this approach, both the objectives of privacy protection and statistically-based rule accuracy can be achieved. The randomized value

distortion technique for learning decision trees [3] and privacy preserving association rule mining [14,15] are examples of this approach.

Recent developments [1] have provided an expectation–maximization (EM) algorithm for reconstructing the distribution of the original data from perturbed data. This approach describes information theoretic measures that quantify the amount of privacy provided by a randomization approach. Furthermore, a random matrix-based spectral filtering technique has been proposed [22] to challenge privacy preserving approaches based on random data perturbation.

In this paper, we present a challenge for privacy preservation when performing data mining techniques on multi-relational databases. We believe that this new viewpoint can both reveal potential security holes and indicate better ways of protecting database security.

4 Basic concepts of hyperclique patterns

The hyperclique pattern was the inspiration for our pattern-based semi-supervised learning approach, and thus, the pattern that we use to explore this idea. In this section, we describe the concept of hyperclique patterns. Note that although the hyperclique pattern is described as a set of items, as is traditional for association patterns, if the data matrix is transposed, the hypercliques that are then found consist of sets of transactions (objects). It is hypercliques of objects that are used in this paper.

4.1 Hyperclique patterns

A hyperclique pattern is a type of association pattern that contains items that are *highly affiliated* with each other. By high affiliation, we mean that the presence of an item in a transaction strongly implies the presence of every other item that belongs to the same pattern. The h-confidence measure [33] is specifically designed to capture the strength of this association.

Definition 1 *The* **h-confidence** *of a pattern* $P = \{i_1, i_2, \ldots, i_m\}$*, denoted as* $\text{hconf}(P)$*, is a measure that reflects the overall affinity among items within the pattern. This measure is defined as* $\min(\text{conf}(\{i_1\} \rightarrow \{i_2, \ldots, i_m\}), \text{conf}(\{i_2\} \rightarrow \{i_1, i_3, \ldots, i_m\}), \ldots, \text{conf}(\{i_m\} \rightarrow \{i_1, \ldots, i_{m-1}\}))$*, where* conf *is the classic definition of association rule confidence* [2].

Example 3 Let us consider an itemset $P = \{A, B, C\}$. Assume that $\text{supp}(\{A\}) = 0.1$, $\text{supp}(\{B\}) = 0.1$, $\text{supp}(\{C\}) = 0.06$, and $\text{supp}(\{A, B, C\}) = 0.06$, where supp is the classic definition of association rule support [2]. Then

$$\text{conf}(\{A\} \rightarrow \{B, C\}) = \text{supp}(\{A, B, C\})/\text{supp}(\{A\}) = 0.6$$
$$\text{conf}(\{B\} \rightarrow \{A, C\}) = \text{supp}(\{A, B, C\})/\text{supp}(\{B\}) = 0.6$$
$$\text{conf}(\{C\} \rightarrow \{A, B\}) = \text{supp}(\{A, B, C\})/\text{supp}(\{C\}) = 1$$

Therefore, $\text{hconf}(P) = \min(\text{conf}(\{B\} \rightarrow \{A, C\}), \text{conf}(\{A\} \rightarrow \{B, C\}), \text{conf}(\{C\} \rightarrow \{A, B\})) = 0.6$.

Table 2 Examples of hyperclique patterns from the LA1data set

LA1 dataset		
Hyperclique patterns	Support	H-confidence
{gorbachev, mikhail}	1.4%	93.6%
{photo, graphic, writer}	14.5%	42.1%
{sentence, convict, prison}	1.4%	32.4%
{rebound, score, basketball}	3.8%	40.2%
{season, team, game, play}	7.1%	31.4%

Definition 2 *A pattern* P *is a* **hyperclique pattern** *if* $\text{hconf}(P) \geq h_c$*, where* h_c *is a user-specified minimum h-confidence threshold. A hyperclique pattern is a* **maximal hyperclique pattern** *if no superset of this pattern is also a hyperclique pattern.*

Table 2 shows some hyperclique patterns identified from words of the LA1 dataset, which is part of the TREC-5 collection [31] and includes articles from various news categories such as 'financial,' 'foreign,' 'metro,' 'sports,' and 'entertainment.' For instance, in Table 2, the hyperclique pattern {season, team, game, play} is from the 'sports' category.

4.2 Properties of the H-confidence measure

The h-confidence measure has three important properties, namely the anti-monotone property, the cross-support property, and the strong affinity property. Detailed descriptions of these three properties were provided in our earlier paper [33]. Here, we provide only the following brief summaries.

The anti-monotone property guarantees that if an itemset $\{i_1, \ldots, i_m\}$ has an h-confidence value of h_c, then every subset of size $m - 1$ also has an h-confidence value of h_c. This property is analogous to the anti-monotone property of support used in association-rule mining [2] and allows us to use h-confidence-based pruning to speed the search for hyperclique patterns in the same way that support-based pruning is used to speed the search for frequent itemsets.

The cross-support property provides an upper bound for the h-confidence of itemsets that contain items from different levels of support. The computation of this upper bound is much cheaper than the computation of the

exact h-confidence value, since it only relies on the support values of individual items in the itemset. Using this property, we can design a partition-based approach that allows us to efficiently eliminate patterns involving items with different support levels.

The strong affinity property guarantees that if a hyperclique pattern has an h-confidence value above the minimum h-confidence threshold, h_c, then every pair of items within the hyperclique pattern must have a cosine similarity (uncentered Pearson correlation coefficient) greater than or equal to h_c. As a result, the overall affinity of hyperclique patterns can be controlled by setting an h-confidence threshold.

The anti-monotone and cross-support properties form the basis of an efficient hyperclique mining algorithm that has much better performance than traditional frequent pattern mining algorithms, particularly at low levels of support [33].

5 Semi-supervised learning using nearest neighbor approaches

In general, there are two potential approaches for semi-supervised learning using nearest neighbors. The first one is a K nearest neighbor based semi-supervised (KNNS) learning method. The second one is a Top-k nearest neighbor based semi-supervised (TOP-K NNS) learning method.

5.1 The KNNS method

We first describe the KNNS method. For each given object with a class label, the KNNS method uses the class label of the given object to label each of its k nearest neighbors. If a predicted object is found to be one of k nearest neighbors of more than one given object, then the KNNS method assigns the label of the given object with the highest similarity.

The KNNS method has several desirable characteristics. First, the method is simple and easy to implement. Second, it classifies only those objects that are near neighbors of objects with known class labels, thus potentially achieving higher accuracy than methods which attempt to predict class labels for all objects.

However, the KNNS method only considers pairs of similar objects when labeling the data objects. Indeed, in real-world data sets, it is possible that two objects are often nearest neighbors without belonging to the same class [29]. This is also illustrated by experimental results, as shown in Sect. 8.1. Thus, by looking at only pairwise similarity, the KNNS method tends to guarantee a certain number of errors in many data sets.

In addition, the KNNS method predicts an equal number of objects for each object with a class label; that is, this method gives each labeled object equal weight as a predictor. This may not be appropriate in real-world data sets. Intuitively, objects from a high-density cluster may predict more objects with a high accuracy. In contrast, objects from a loosely connected cluster may only have limited predictive power. In the worst case, a labeled object can be noise or an outlier that is completely unsuitable for prediction.

5.2 The TOP-K NNS method

In this subsection, we present the TOP-K NNS method. For n given objects with class labels, the TOP-K NNS method finds the k objects with the highest level of similarity from the neighborhood of these n objects.

The TOP-K NNS method also has several appealing characteristics. First, this method is simple and is easy to implement. Second, like the KNNS method, the TOP-K NNS method only classifies objects that are near neighbors of objects with known class labels. Finally, based on similarity, the TOP-K NNS method assigns different predictive power to different labeled objects. As a result, unlike the KNNS method, the TOP-K NNS method can avoid many prediction errors when some labeled objects are noise or outliers.

Like the KNNS, the prediction mechanism of the TOP-K NNS method is also solely based on pairwise similarity. As already noted, in real-world data sets, it is possible that two objects can be nearest neighbors without belonging to the same class. Therefore, TOP-K NNS can also generate many prediction errors.

Furthermore, the TOP-K NNS method gives higher predictive power to objects from a dense cluster. If the labeled objects are from both dense and sparse clusters, it is possible that (1) few objects from a sparse cluster can be predicted if the value of k is small and (2) if a large value of k is specified, then some objects from a sparse cluster can be predicted. However, in the latter case, more prediction errors will be introduced for those objects from dense clusters.

6 HPSL: hyperclique pattern based semi-supervised learning

In this section, we propose a hyperclique pattern based semi-supervised learning (HPSL) method.

6.1 The HPSL algorithm

Our formulation of the problem of privacy leakage in multi-relational databases assumes that only a small

number of the objects have class labels. For an object with a class label, our purpose is to find a maximal hyperclique pattern that contains this object and then label all other objects in the pattern with the label of this object. However, there are two special cases that require extra processing. First, if the hyperclique pattern contains objects with different class labels, then our algorithm assigns an unlabeled object the class label of the labeled object that has the highest similarity to the unlabeled object. A similar strategy can be applied for the second case, where an unlabeled object is located in two different maximal hyperclique patterns. In practice, these special cases do not occur frequently.

Figure 2 shows the pseudocode of the HPSL algorithm. This algorithm consists of two phases. In the first phase, the HPSL finds maximal hyperclique patterns that contain at least one given object O with a class label. Note that we implement this on top of an efficient maximal hyperclique pattern mining algorithm [20] by pushing the constraint that a hyperclique pattern must contain a labeled object into the pattern discovery process. In the second phase, the HPSL labels all the unlabeled objects in the discovered maximal hyperclique patterns.

There are several benefits of the HPSL method. First, this method only predicts class labels for objects strongly connected to objects with known class labels. Recall that the KNNS and TOP-K NNS methods also have this characteristic. Second, unlike the KNNS method, the HPSL considers the similarity among groups of objects instead of just pairs of objects. Third, hyperclique patterns represent unique concepts that may potentially help guide better information inference in databases. Finally, the application of the HPSL method for attacking database security reveals an interesting direction for multi-relational data mining [11].

HPSL Algorithm

Input: D: a data set.
O: a set of objects with class labels.
θ: a minimum h-confidence threshold.
α: a minimum support threshold.
HS: a set to store hyperclique patterns

Output: SLR: Semi-supervised Learning Results.

Method

Phase I: Hyperclique Mining with Object Constraints

1. HS ← maximal_hyperclique(D, O, θ, α)

Phase II: Semi-supervised Learning

2. **for** each pattern p in HS
3. SLR ← SLR ∪ semi_label(p)
4. **endfor**
5. OUTPUT SLR
6. **End**

Fig. 2 The HPSL algorithm

6.2 Comparison of the HPSL method and the KNNS method

We illustrate the difference between the KNNS method and the HPSL method in terms of the scope of objects that may be predicted. Assume that we have already known class labels for five objects: O_1, O_2, O_3, O_4, and O_5. The KNNS method will find the k most similar neighbors around each object with a class label and label these neighbors using the label of the given object. In this case, the KNNS method treats every object with a class label as an equal predictor and will predict the same number of objects for each labeled object. In contrast, for each object with a class label, the HPSL method finds the maximal hyperclique pattern that contains this object and labels all members of this pattern with this class label. Hence, for the HPSL method, each object with a class label has different predictive power. In other words, different numbers of objects are predicted for each object with a class label. This is desirable since it better reflects reality.

The rationale behind the HPSL method is as follows. If there is a clustering effect in a real life data set, then different clusters often have different cluster sizes. Objects representing noise and outliers may also be present. Hence, it is natural that different objects should have different predictive power, since an object that represents noise or an outlier may not predict anything, while an object from a large cluster may be used to predict class labels for many objects within this cluster. Consider the following example.

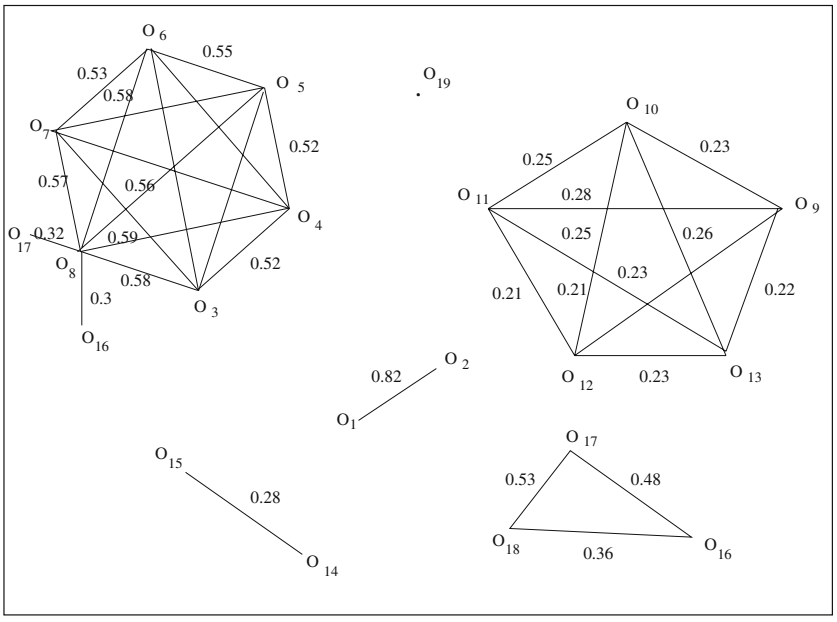


Fig. 3 The working mechanism of the HPSL method

Example 4 Figure 3 shows a hyperclique pattern graph of a sample data set with 19 objects at the h-confidence threshold 0.2. If the cosine similarity of two objects is above 0.2, an edge is drawn between them and the similarity value is the weight of the edge.[3] In this sample data set, we assume that the class labels for objects, O_1, O_3, and O_{19}, are known. If the KNNS method is applied, k nearest neighbors around O_1, O_3, and O_{19} will be found and labeled using the class label of O_1, O_3, and O_{19}, respectively. Not surprisingly, the prediction accuracy is extremely poor. Since the object O_{19} is a noise point, the class label of O_{19} is expected to be different from its nearest neighbors, but they will be assigned the class label of O_{19} by the KNNS method. Furthermore, for the object O_3, the KNNS method makes a prediction for its k similar neighbors; however, the scope of this inference is limited. In contrast, the HPSL method will first find a maximal hyperclique pattern for each object with a class label. Since there is no hyperclique pattern that can be found for the object O_{19}, no objects can be predicted by this object using the HPSL method. Also, since $\{O_3, O_4, O_5, O_6, O_7, O_8\}$ is a hyperclique pattern at the h-confidence threshold 0.2 – recall that the cosine similarity of pairs of objects in this pattern is above 0.2 – the HPSL method will label all members in this pattern with the label of the object O_3. Hence, the HPSL method should achieve better prediction accuracy and be more suitable for real-world situations.

[3] To avoid clutter, not all edges are labeled

6.3 A comparison between the HPSL method and the TOP-K NNS method

The major difference between the HPSL method and the TOP-K NNS method is that the TOP-K NNS method is solely based on pairwise similarity. In contrast, the HPSL takes the similarity among all objects in a hyperclique into consideration, rather than relying only on pairwise similarity. Therefore, the HPSL has the potential for avoiding prediction errors that result from the pairwise similarity approach. Furthermore, if the labeled objects come from clusters with different cluster densities, it is possible that k objects with top k highest similarity are all from a dense cluster. Hence, no object will be predicted from a cluster with a lower density even though objects in a cluster with a lower density may be tightly coupled. Consider the following example.

Example 5 For the data set in Fig. 3, assume that the class labels of objects O_{11} and O_8 are known and we want to predict five objects. For the TOP-K NNS method, five objects, O_3, O_4, O_5, O_6, O_7, around O_8 have the top-5 highest similarity, so these objects are labeled by the label of O_8 and no object will be predicted by the object O_{11}. In order to predict some objects around O_{11}, we have to increase the number of objects for prediction. However, this may result in some prediction errors around the labeled object O_8 due to the use of pairwise similarity. For instance, before we can predict objects around O_{11}, we must first predict additional objects, such as O_{16} and O_{17}, around the object O_8. This can introduce many prediction errors. In contrast,

the HPSL method can find $\{O_9, O_{10}, O_{11}, O_{12}, O_{13}\}$ and $\{O_3, O_4, O_5, O_6, O_7, O_8\}$ as hyperclique patterns and then label all members in these two patterns with the labels of O_8 and O_{11}, respectively.

7 A prevention principle

The results of security breaches in database views can be extremely serious. In this section, we investigate how to prevent potential information leakage from semi-supervised learning in multi-relational databases.

In the following, we first give a principle which can be used to guide the design of effective techniques for preventing semi-supervised learning attacks.

Principle 1 *A prevention technique should have the ability to decluster the data, i.e., similar data objects tend to have different class labels.*

The rationale of this principle follows directly from the hypothesis of semi-supervised learning attacks: similar data objects should have similar class labels. Following this principle, we provide a preventive technique that introduces pseudo-attributes into the database. The purpose of these pseudo-attributes is to make sure that the objects with the same class label have poor similarity to each other. In other words, pseudo-attributes can decluster objects with the same class label.

The key idea of the pseudo-attribute perturbation method is illustrated as follows. First, consider Eq. 1 which defines the cosine similarity measure for two vectors $x = \{x_1, x_2, \ldots, x_m\}$ and $y = \{y_1, y_2, \ldots, y_m\}$. Our purpose is to introduce some pseudo-attributes into these two vectors such that the value $\sum x_i y_i$ is decreased and the value $\sqrt{\sum x_i^2 \sum y_i^2}$ is increased. As a result, the cosine similarity between x and y is reduced.

$$\cos(x, y) = \frac{\sum x_i y_i}{\sqrt{\sum x_i^2 \sum y_i^2}} \tag{1}$$

Figure 4 shows the pseudo-attribute perturbation algorithm. Line 1 randomly samples k attributes from the data set for perturbation. For the data set with the selected k attributes, Line 2 partitions objects into p groups based on their class labels. For each group, Line 4 computes the centroid vector **c** and Line 6 reduces the cosine similarity between each object O and the centroid **c** using the abovementioned idea. The data with perturbed attributes are combined with the original data set to form a new data set.

Finally, we should point out that the earlier proposed preventive technique may not be the best approach to deter semi-supervised learning attacks. However, it does increase the complexity for an attacker to a higher level. Furthermore, since this approach only introduces pseudo-attributes into the database, a query on the original data will not be affected.

Pseudo-Attribute Perturbation

Input: D: a data set.
m: the number of attributes.
n: the number of data objects.
p: the number of classes.

Variables: S: a set of pseudo-attributes.

Output: an extended data set D′ with k pseudo-attributes

Method

1. S ← random_sample_attributes(k, D)
2. $(S_1, S_2, \ldots, S_p)$ ← partition(S, p)
3. **For** i = 1 to p do
4. **c** ← centroid(S_i)
5. **For** each O in S_i do
6. O ← perturbation(O, **c**)
7. End for
8. **End for**
9. S ← $(S_1, S_2, \ldots, S_p)$
10. D′ = D ∪ S
11. OUTPUT D′
12. **End**

Fig. 4 Pseudo-attribute perturbation

8 Experimental evaluation

In this section, we demonstrate the information leakage in databases via the HPSL method with experiments on

several real-world data sets. The relative performance between HPSL and KNNS, as well as TOP-K NNS, is also presented. To conclude this section, we present results that show the effectiveness of the pseudo-attribute perturbation technique.

8.1 Experimental setup

Experimental data sets For our experiments, we used three real-world data sets. Some characteristics of these data sets are shown in Table 3. The LA1 data set is part of the TREC-5 collection [31] and contains news articles from the Los Angeles Times. The RE0 data set is from the Reuters-21578 text categorization test collection Distribution 1.0 [24]. The data set WAP is from the WebACE project [19]; each document corresponds to a web page listed in the subject hierarchy of Yahoo! (http://www.yahoo.com). For all data sets, we used a stop-list to remove common words, and the words were stemmed using Porter's suffix-stripping algorithm [26].

Entropy measure In our experiments, we applied the entropy measure for evaluating the clustering effect in a data set. To compute the entropy of a set of clusters, we first calculate the class distribution of the objects in each cluster, i.e., for each cluster j we compute p_{ij}, the probability that a member of cluster j belongs to class i. Given this class distribution, the entropy, E_j, of cluster j is calculated using the standard entropy formula

$$E_j = -\sum_i p_{ij} \log(p_{ij}), \tag{2}$$

where the sum is taken over all classes and the log is log base 2. The total entropy for a set of clusters is computed as the weighted sum of the entropies of each cluster, as shown in the equation

$$E = \sum_{j=1}^{m} \frac{n_j}{n} * E_j, \tag{3}$$

where n_j is the size of cluster j, m is the number of clusters, and n is the total number of data points.

Table 3 Characteristics of real-world document data sets

Data set	LA1	RE0	WAP
Number of documents	3204	1504	1560
Number of words	31472	11465	8460
Number of classes	6	13	20
Min class size	273	11	5
Max class size	943	608	341
Min/max class size	0.29	0.018	0.015
Source	TREC-5	Reuters	WebAce

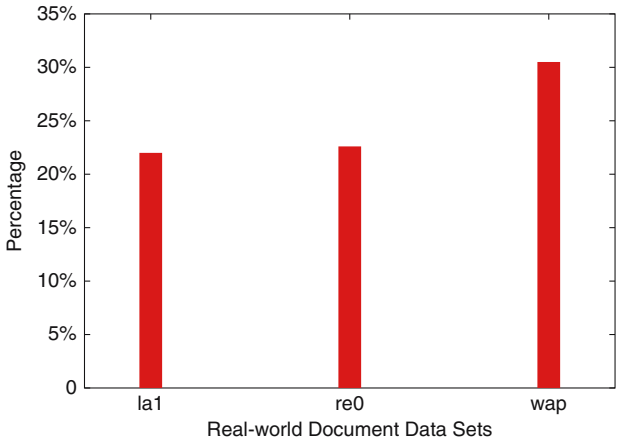


Fig. 5 The percent of documents whose nearest neighbor is of a different class

8.2 Problems with NN approaches

In real-world data sets, while objects can display a clustering effect globally, two objects can often be nearest neighbors without belonging to the same class. To illustrate this, let us consider real-world document data sets. Figure 5 shows the percent of documents whose nearest neighbor is not of the same class. While this percentage varies widely from one data set to another, the chart confirms what we have just stated about nearest neighbor behavior in document data sets.

Since, in many cases, the nearest neighbors of an object have a different class than the object, nearest neighbor-based semi-supervised learning approaches will often assign objects of different classes to the same class. To cope with this challenge, our HPSL method considers the similarity among all objects in a hyperclique pattern instead of only the two most similar objects.

8.3 Cluster nature of hyperclique patterns

In this experiment, we explain why the hyperclique pattern is a good candidate for pattern-based semi-supervised learning. Figure 6 shows, for the LA1 data set, the entropy of the discovered hyperclique patterns for different minimum h-confidence and support thresholds. Note that when the minimum h-confidence threshold is zero, we actually have frequent itemset patterns instead of hyperclique patterns. As Fig. 6 shows, when the minimum h-confidence threshold increases, the entropy of hyperclique patterns decreases dramatically. For instance, when the h-confidence threshold is higher than 0.25, the entropy of hyperclique patterns will be less than 0.1 at all the given minimum support thresholds. This indicates that, at certain h-confidence thresholds,

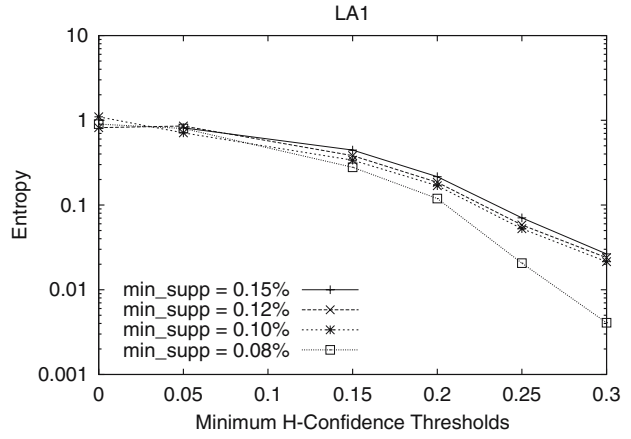


Fig. 6 Illustration of the cluster nature of hyperclique patterns on the LA1 document data set

hyperclique patterns tend to include objects from the same class. In contrast, the entropy of frequent patterns is high – close to 1 – for all the given minimum support thresholds. This means that frequent patterns tend to include objects from different classes. Thus, with respect to purity, the hyperclique pattern is a better candidate than frequent patterns for pattern-based semi-supervised learning.

Another trend that can be observed in Fig. 6 is that, with the decrease of the minimum support thresholds, the entropy of hyperclique patterns from the LA1 data set trends downward. This indicates that high affinity patterns can appear at very low levels of support. However, frequent itemset mining algorithms have difficulty in identifying frequent itemsets at low levels of support. In contrast, the hyperclique pattern mining algorithm has much better performance at low levels of support [33]. Thus, if we want to discover high-affinity patterns occurring at low levels of support, then the hyperclique pattern is a better choice.

8.4 The effect of changing the number of objects with known class labels

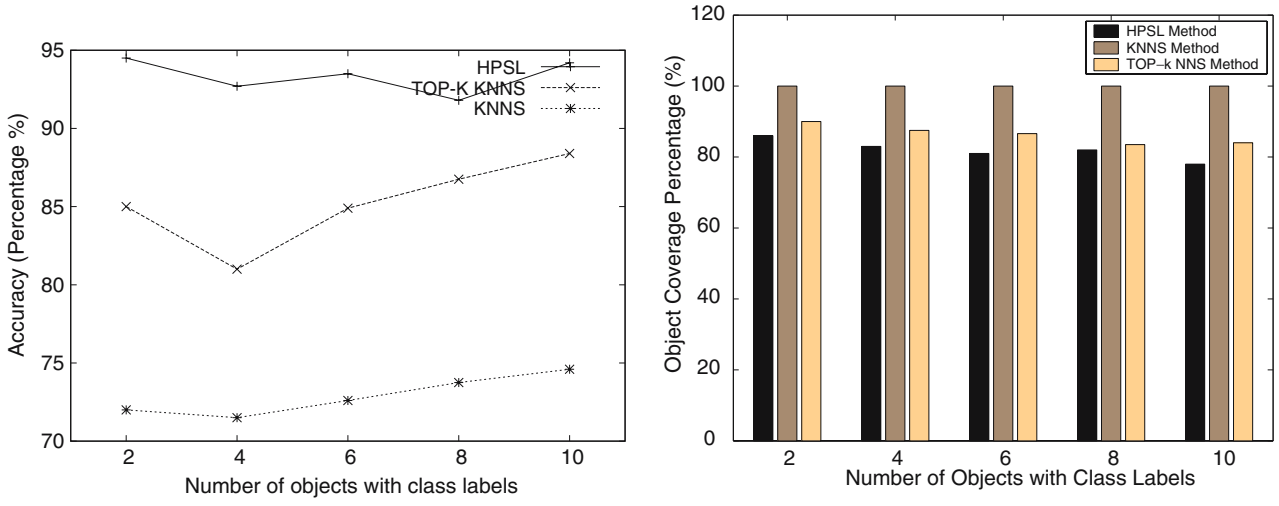
Here, we show the relative performance of the HPSL method and the nearest neighbor-based approaches, KNNS and TOP-K NNS, as the number of objects with known class labels is increased. More specifically, we present the prediction accuracy and the object coverage – the percentage of the given labeled objects that have been used for prediction – when the number of objects with class labels is 2, 4, 6, 8, and 10, respectively (we stress that 'object coverage' is not the percentage of objects labeled, just the percentage of labeled objects used for prediction). The KNNS always uses all labeled objects for classification, but the HPSL and TOP-K NNS may not. In this experiment, we specify the total number of predicted objects to be approximately five times more than the number of objects with class labels. We also performed random sampling to select objects with class labels. Finally, in order to reduce effect of random fluctuations, we conducted 10 trials for each experiment.

Figures 7a, 8a, and 9a show the classification accuracy of HPSL, KNNS, and TOP-K NNS on the WAP, LA1, and RE0 data sets, respectively. As can be seen, for most observed numbers of objects with known class labels, the achieved accuracy of the HPSL method is significantly and systematically better than that of the KNNS and TOP-K NNS methods. This is due to the fact that the HPSL method has the power to eliminate the isolated data objects that often result in prediction errors in nearest neighbor approaches, such as the KNNS and TOP-K NNS. Another observation is that the TOP-K NNS method performs much better than the KNNS in terms of accuracy. Indeed, the KNNS method is forced to predict the same number of objects for each labeled object. If objects that are a noise point or an outlier are picked, the KNNS method tends to make a wrong prediction. In contrast, the TOP-K NNS method only predicts objects with the top k highest similarity. Hence, it is possible that no prediction will be made for points that are a noise point or an outlier, thus reducing the chance of incorrect predictions.

Figures 7b, 8b, and 9b show the object coverage percentage by the HPSL, KNNS, and TOP-K NNS on WAP, LA1, and RE0 data sets, respectively. Since the KNNS method is forced to use each labeled object for prediction, the object coverage of KNNS is always 100%. Furthermore, we observed that the object coverage of HPSL is slightly smaller than that of TOP-K NNS.

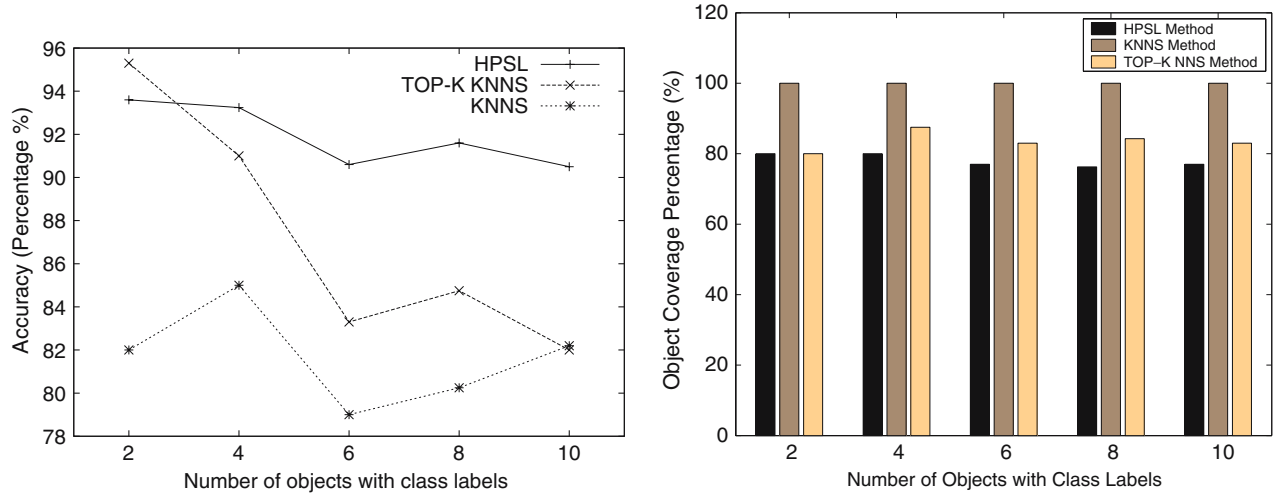
8.5 The effect of changing the proportion of predicted objects

In this subsection, we show the effect of changing the proportion of predicted objects on the performance of HPSL, KNNS, and TOP-K NNS. In the experiment, we set the number of objects with class labels to be six and performed a random sampling to select these six objects. Furthermore, 10 trials were conducted for each observed parameter. Figure 10 shows the prediction accuracy of HPSL, KNNS, and TOP-K NNS on the RE0, LA1, and WAP data sets as the proportion of predicted objects is increased. In the figure, we can observe that the accuracy of the HPSL method is much better that that of the KNNS or TOP-K methods on all three testing data sets. Furthermore, for all three methods, accuracy trends downward when the proportion of estimated instances is increased. This result is not surprising, since we would



a) WAP: Accuracy **b)** WAP: Object Coverage

Fig. 7 The effect of changing the number of objects with known class labels on the WAP data set in terms of the prediction accuracy and the object coverage



a) LA1: Accuracy **b)** LA1: Object Coverage

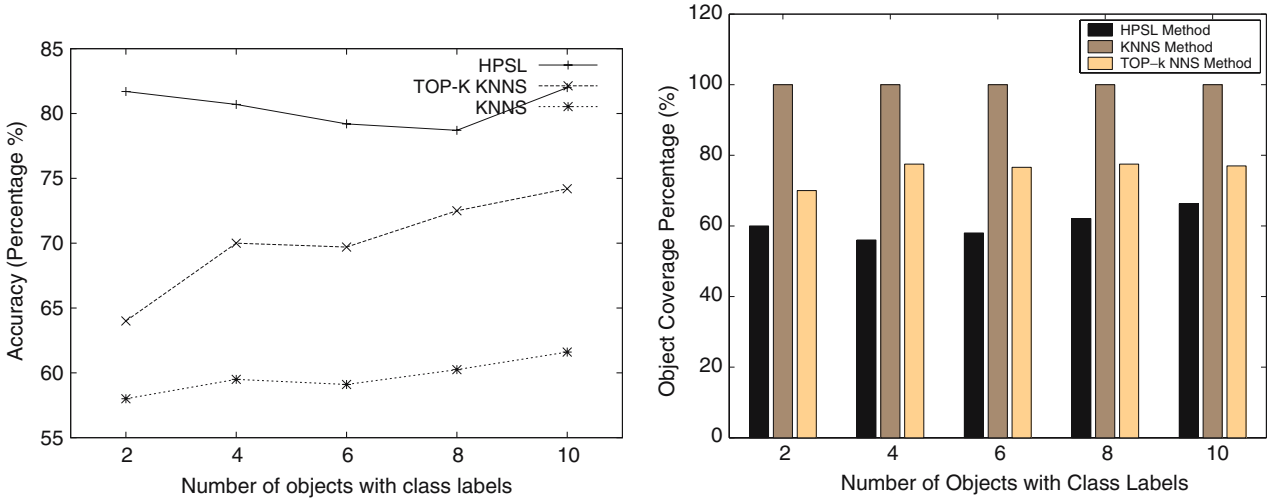
Fig. 8 The effect of changing the number of objects with known class labels on the LA1 data set in terms of the prediction accuracy and the object coverage

expect that predictability would decrease when the similarity of objects decreases.

8.6 The effectiveness of the pseudo-attribute perturbation technique

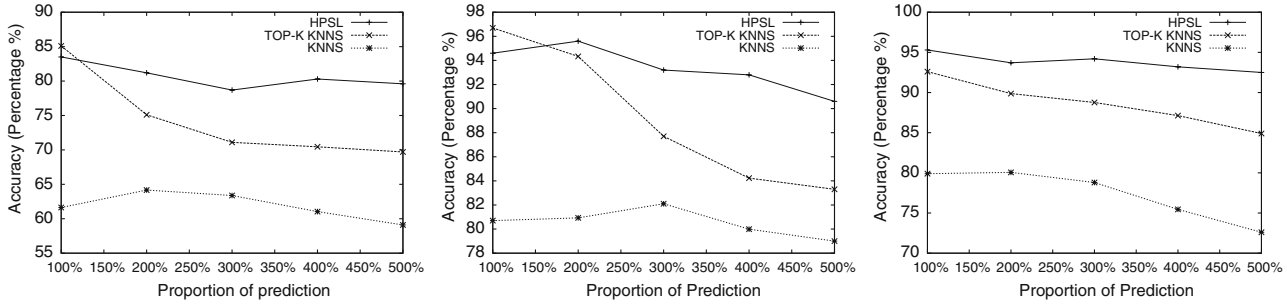
In this experiment, we test the effectiveness of the pseudo-attribute perturbation technique on three real life data sets: RE0, LA1, and WAP. More specifically, our purpose is to show that the pseudo-attribute perturbation technique can introduce the declustering effect among objects in the same category. In other words, this preventive technique tends to invalidate the basic assumption on which semi-supervised learning attacks on multi-relational databases are based.

Figure 11 shows the entropy of sets of clusters found by the bisecting K-means clustering method [23] before and after the pseudo-attribute perturbation. As can be seen, if no pseudo-attribute perturbation is involved, the entropy is relatively low for different sizes of clusters on all three data sets. This indicates that there is a strong clustering effect in each category for all three test data sets. After we applied the pseudo-attribute perturbation technique, the entropy jumps significantly for all three test data sets, i.e., there is a declustering effect on objects within the same category.



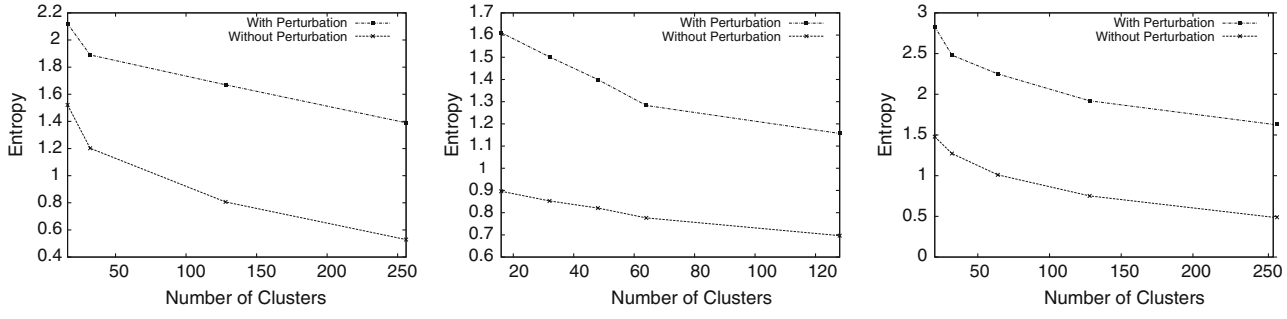
a) RE0: Accuracy **b)** RE0: Object Coverage

Fig. 9 The effect of changing the number of objects with known class labels on the RE0 data set in terms of the prediction accuracy and the object coverage



(a) *RE0* **(b)** *LA1* **(c)** *WAP*

Fig. 10 The effectiveness of changing the proportion of predicted objects on the RE0, LA1, and WAP data sets



(a) *RE0* **(b)** *LA1* **(c)** *WAP*

Fig. 11 The effectiveness of the pseudo-attribute perturbation for the RE0, LA1, and WAP data sets

9 Conclusions

In this paper, we demonstrated a new challenge for database security from the data mining perspective. More specifically, we presented a framework for illustrating potential privacy leakage in multi-relational databases via semi-supervised learning. The hypothesis behind this framework is that similar data objects tend to have similar class labels. We also introduced a new semi-supervised learning approach, the HPSL. This method differs from the traditional nearest neighbor method, since it considers the similarity among groups of objects instead of only pairs of objects. As demonstrated by our experimental results, the HPSL method can achieve

better prediction accuracy than the traditional KNNS and TOP-K NNS approaches. Finally, we proposed a principle for protecting multi-relational databases from such semi-supervised learning attacks. A simple pseudo-attribute perturbation method was also provided. Our experiments showed that the pseudo-attribute perturbation method can reduce the clustering effect in data sets, thus reducing the risk of semi-supervised learning attacks.

For future work, we plan to investigate other semi-supervised learning techniques from machine learning, data mining, statistics, or other domains, such as clique-finding techniques or clustering based on the complete link algorithm. We also plan to investigate privacy leakage for other application data, such as medical data. Finally, we are interested in developing a quantitative approach to estimate how much privacy can be maintained under a semi-supervised learning attack.

Acknowledgements This work was partially supported by NSF grant no. IIS-0308264, NSF ITR no. 0325949, and by the Army High Performance Computing Research Center under the auspices of the Department of the Army, Army Research Laboratory cooperative agreement number DAAD19-01-2-0014. The content of this work does not necessarily reflect the position or policy of the government and no official endorsement should be inferred. Access to computing facilities was provided by the AHPCRC and the Minnesota Supercomputing Institute. Finally, we would like to thank the editors of the special issue, Elena Ferrari and Bhavani Thuraisingham, as well as our anonymous reviewers for their tremendous effort in helping us develop this paper and for an exemplary review process as a whole.

References

1. Agrawal, D., Aggarwal, C.C.: On the design and quantification of privacy preserving data mining algorithms. In: Proceedings of the ACM Symposium on Principles of Database Systems (PODS) (2001)
2. Agrawal, R., Imielinski, T., Swami, A.: Mining association rules between sets of items in large databases. In: Proceedings of the ACM SIGMOD Conference on Management of Data (1993)
3. Agrawal, R., Srikant, R.: Privacy-preserving data mining. In: Proceedings of the ACM SIGMOD Conference on Management of Data (2000)
4. Bayardo, R.J., Srikant, R.: Technological solutions for protecting privacy. In: IEEE Computer (2003)
5. Bertino, E., Ooi, B.C., Yang, Y., Deng, R.H.: Privacy and ownership preserving of outsourced medical data. In: Proceedings of the 21st International Conference on Data Engineering (ICDE), pp. 521–532 (2005)
6. Carminati, B., Ferrari, E., Bertino, E.: Assuring security properties in third-party architectures. In: Proceedings of the 21st International Conference on Data Engineering (ICDE), pp. 547–548 (2005)
7. Castelli, V., Cover, T.M: The relative value of labeled and unlabeled samples in pattern recognition with an unknown mixing parameter. IEEE Trans. Inf. Theory, **42**(6), 2102–2117 (1996)
8. Codd, E.: A relational model for large shared data banks. Comm. ACM **13**(6), 377–387 (1970)
9. Denning, D., Akl, S., Morgenstern, M., Neumanna, P.: Views for multilevel database security. In: IEEE Symposium on Security and Privacy (1986)
10. Denning, D., Lunt, T., Schell, R., Heckman, M., Shockley, W.: Views for multilevel database security. In: IEEE Symposium on Security and Privacy (1986)
11. Domingos, P.: Prospects and challenges for multi-relational data mining. SIGKDD explorations (2003)
12. Du, W., Han, Y.S., Chen, S.: Privacy-preserving multivariate statistical analysis: linear regression and classification. In: Proceedings of the 4th SIAM International Conference on Data Mining (2004)
13. Duin, R.: Classifiers in almost empty spaces. In: Proceedings of 15th International Conference on Pattern Recognition (2000)
14. Evfimievski, A., Gehrke, J., Srikant, R.: Limiting privacy breaches in privacy preserving data mining. In: Proceedings of the 22nd ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems (2002)
15. Evfimievski, A., Srikant, R., Agrawal, R., Gehrke, J.: Privacy preserving mining of association rules. In: Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (2002)
16. Faloutsos, C., Jagadish, H.V., Sidiropoulos, N.: Recovering information from summary data. In: Proceedings of 23rd International Conference on Very Large Data Bases (VLDB), pp. 36–45 (1997)
17. Ferrari, E., Thuraisingham, B.M.: Security and privacy for web databases and services. In: Proceedings of the 9th International Conference on Extending Database Technology (EDBT), pp. 17–28 (2004)
18. Ghahramani, Z., Jordan, M.I.: Supervised learning from incomplete data via an EM approach. In: NIPS, pp. 120–127 (1993)
19. Han, E.-H., Boley, D., Gini, M., Gross, R., Hastings, K., Karypis, G., Kumar, V., Mobasher, B., Moore, J.: Webace: a web agent for document categorization and exploration. In: Proceedings of the 2nd International Conference on Autonomous Agents (1998)
20. Huang, Y., Xiong, H., Wu, W., Zhang, Z.: A hybrid approach for mining maixmal hyperclique patterns. In: ICTAI, pp. 354–361 (2004)
21. Huang, Z., Du, W., Chen, B.: Deriving private information from randomized data. In: Proceedings of the ACM SIGMOD Conference, pp. 37–48 (2005)
22. Kargupta, H., Datta, S., Wang, Q., Sivakumar, K.: On the privacy preserving properties of random data perturbation techniques. In: Proceedings of the 3rd IEEE International Conference on Data Mining, pp. 387–394 (2003)
23. Karypis, G.: Cluto: Software for clustering high dimensional datasets. /www.cs.umn.edu/∼karypis
24. Lewis, D.: Reuters-21578 text categorization text collection 1.0. In: http://www.research.att.com/ lewis
25. Nigam, K., McCallum, A., Thrun, S., Mitchell, T.M.: Text classification from labeled and unlabeled documents using EM. Mach. Learn. **39**(2/3), 103–134 (2000)
26. Porter, M.F.: An algorithm for suffix stripping. In: Program, **14**(3), (1980)
27. Raudys, S., Jain, A.: Small sample size effects in statistical pattern recognition: Recommendations for practitioners. IEEE Trans. Pattern Anal. Mach. Intell. **13**(3), 252–264 (1991)
28. Seeger, M.: Learning with labeled and unlabeled data. In: Technical Report, University of Edinburgh (2001)
29. Steinbach, M., Karypis, G., Kumar, V.: A comparison of document clustering techniques. In: KDD Workshop on Text Mining (2000)

30. Steinbach, M., Tan, P.N., Xiong, H., Kumar, V.: Generalizing the notion of support. In: Proceedings of the 2004 ACM SIGKDD Int'l Conference on Knowledge Discovery and Data Mining pp. 689–694. ACM Press (2004)
31. TREC.: In: http://trec.nist.gov.
32. Xiong, H., Steinbach, M., Kumar, V.: Privacy leakage in multi-relational databases via pattern based semi-supervised learning. In: Proceedings of the ACM Conference on information and Knowledge Management (CIKM) (2005)
33. Xiong, H., Tan, P., Kumar, V.: Mining strong affinity association patterns in data sets with skewed support distribution. In: Proceedings of the third IEEE International Conference on Data Mining (ICDM), pp. 387–394 (2003)