On Differentially Private Inductive Logic Programming

Chen Zeng, Eric Lantz, Jeffrey F. Naughton, David Page

Department of Computer Sciences, University of Wisconsin-Madison, USA
{zeng,lantz,naughton}@cs.wisc.edu, page@biostat.wisc.edu

Abstract. We consider differentially private inductive logic programming. We begin by formulating the problem of guaranteeing differential privacy to inductive logic programming, and then prove the theoretical difficulty of simultaneously providing good utility and good privacy in this task. While our analysis proves that in general this is very difficult, it leaves a glimmer of hope in that when the size of the training data is large or the search tree for hypotheses is "short" and "narrow," we might be able to get meaningful results. To prove our intuition, we implement a differentially private version of Aleph, and our experimental results show that our algorithm is able to produce accurate results for those two cases.

1 Introduction

Recently, concomitant with the increasing ability to collect personal data, privacy has become a major concern. In this paper, we focus on privacy issues that arise in the context of inductive logic programming (ILP).

Given an encoding of a set of examples represented as a logical database of facts, an ILP algorithm will attempt to derive a hypothesized logic program which entails all the positive and none of the negative examples. Developing efficient algorithms for ILP has been widely studied by the machine learning community [1]. However, to the best of our knowledge, a differentially private approach to ILP has not received any attention.

ILP induces hypotheses from examples collected from individuals and synthesizing new knowledge from the examples. This approach naturally creates a privacy concern — how can we be confident that publishing these hypotheses and knowledge does not violate the privacy of the individuals whose data are being studied? This problem is compounded by the fact that we may not even know what data the individuals would like to protect nor what side information might be possessed by an adversary. These compounding factors are exactly the ones addressed by *differential privacy* [2], which intuitively guarantees that the presence of an individual's data in a dataset does not reveal much about that individual. Differential privacy has previously been explored in the context of other machine learning algorithms [3] [4] [5]. Accordingly, in this paper we explore the possibility of developing differentially private ILP algorithms. Our

goal is to guarantee differential privacy without obliterating the utility of the algorithm.

An obvious but important observation is that privacy is just one aspect of the problem; utility also matters. In this paper, we quantify the utility of a differentially private ILP algorithm by its likelihood to produce a sound result. Intuitively speaking, "soundness" requires an algorithm to include a hypothesis that is correct in a sufficiently large subset of the database. We start by showing the trade-off between privacy and utility in ILP. Our result unfortunately indicates that the problem is very hard — that is, in general, one cannot simultaneously guarantee high utility and a high degree of privacy.

However, a closer investigation of this negative result reveals that if we can either reduce the hypotheses space or increase the size of the input data, then perhaps there is a differentially private ILP algorithm that is able to produce a high quality result while guaranteeing privacy. To verify this, we implement a differentially private ILP algorithm and run experiments on a synthetic dataset. At the most abstract level, ILP at its heart involves counting the number of rows in the input that satisfies predicates, and we add noise to these counts to ensure privacy. Our results indicate that our algorithm is able to produce results of high quality while guaranteeing differential privacy when those two conditions are met.

The rest of the paper is organized as follows: Section 2 briefly describes the problem of ILP, and the notion of differential privacy. Section 3 formulates the problem of guaranteeing differential privacy to ILP. Section 4 explores the trade-off between privacy and utility in ILP. Section 5 proposes our differentially private ILP algorithm, and Section 6 evaluates our algorithm on a synthetic dataset. Section 8 concludes our work.

2 Preliminaries

In this section we review the problem of ILP and the notion of differential privacy. For ease of exposition our definitions and our theoretical results, which are negative, are presented for the non-monotonic ILP semantics and propositional logic, where examples may be considered as rows or tuples in a single table database. Nevertheless, we also address the standard semantics and multi-table database views in our empirical results in Section 5.

2.1 Inductive Logic Programming

Inductive logic programming investigates the inductive construction of first-order clausal theories from examples. Let $M^+(T)$ be the minimal Herbrand model of a definite clause T. The problem of inductive logic programming is formulated in Definition 1.

Definition 1. *(Inductive logic programming [1]): Given two languages,*

- $\mathcal{L}_1$*: the language of database.*

– $\mathcal{L}_2$*: the language of hypotheses.*

Given a consistent set of background knowledge $D \subseteq \mathcal{L}_1$*, find a hypothesis* $H \in \mathcal{L}_2$*, such that:*

1. *Validity:* $\forall h \in H$*,* h *is true in* $M^+(D)$*.*
2. *Completeness: if general clause* g *is true in* $M^+(D)$*, then* $H \models g$
3. *Minimality: there is no proper subset* G *of* H *which is valid and complete*

In the rest of the paper, we assume both $\mathcal{L}_1$ and $\mathcal{L}_2$ are fixed unless otherwise specified. Note that in the literature of differential privacy [2], the terminology of "background knowledge" is different from the context in ILP and denotes the side information an adversary possesses to attack the privacy of a specific individual in the underlying database. To prevent that confusion, we use the term "database" to represent the background knowledge shown in Definition 1. In the rest of this paper, we use $\|D\|$ to represent the number of rows in a database D. We also refer $\mathcal{L}_2$ as the hypotheses space.

2.2 Differential Privacy

Intuitively, *differential privacy* guarantees that the presence or absence of an individual's information has little effect on the output of an algorithm, and thus, an adversary can learn limited information about any individual. More precisely, for any database $\tau \in D$, let $nbrs(\tau)$ denote the set of *neighboring databases* of τ, each of which differs from τ by at most one *row*. *Differential privacy* requires that the probability of an algorithm to output the same result on any pair of neighboring databases are bounded by a constant ratio.

Definition 2. *(ϵ-differential privacy [2]): For any input database* τ*, a randomized algorithm* f *is* ϵ*-differentially private iff for any* $\mathcal{S} \subseteq Range(f)$*, and any database* $\tau' \in nbrs(\tau)$*,*

$$\Pr(f(\tau) \in S) \leq e^{\epsilon} \Pr(f(\tau') \in \mathcal{S})$$

where Pr *is the probability taken over the coin tosses of the algorithm* f*.*

One way to guarantee differential privacy for a count query is to perturb the correct result. In particular, Ghosh et al. [6] propose the *geometric mechanism* to guarantee ϵ-differential privacy for a single count query. The geometric mechanism adds noise Δ drawn from the two-sided geometric distribution $G(\epsilon)$ with the following probability distribution: for any integer σ,

$$\Pr(\Delta = \sigma) \sim e^{-\epsilon|\sigma|} \tag{1}$$

The geometric mechanism is a discrete variant of the Laplacian mechanism [7], which adds random noise drawn from the Laplacian distribution. To ensure differential privacy for multiple count queries, we first compute the *sensitivity* of those queries, which is the largest difference between the output of those queries on any pair of neighboring databases.

Definition 3. *(Sensitivity): Given d count queries, $\mathbf{q} = \langle q_1, \ldots, q_d \rangle$, the* sensitivity *of* $\mathbf{q}$ *is:*

$$S_{\mathbf{q}} = \max_{\forall \tau, \tau' \in nbrs(\tau)} |\mathbf{q}(\tau) - \mathbf{q}(\tau')|_1$$

Notice that the output of $\mathbf{q}$ is a vector of dimension d, and we use $|\mathbf{x} - \mathbf{y}|_p$ to denote the L_p distance between two vectors $\mathbf{x}$ and $\mathbf{y}$. The following theorem is a straightforward extension of the Laplacian mechanism to the geometric mechanism.

Theorem 1. *Given d count queries $\mathbf{q} = \langle q_1, \ldots, q_d \rangle$, for any database τ, the database access mechanism: $A_{\mathbf{q}}(\tau) = \mathbf{q}(\tau) + \langle \Delta_1, \ldots, \Delta_d \rangle$ where Δ_i is drawn i.i.d from the geometric distribution $G(\epsilon/S_{\mathbf{q}})$ (1), guarantees ϵ-differential privacy for $\mathbf{q}$.*

As proved in [7], a sequence of differentially private computations also ensures differential privacy. This is called the *composition property* of differential privacy as shown in Theorem 2.

Theorem 2. *[7] Given a sequence of computations, denoted as $\mathbf{f} = f_1, \ldots, f_d$, if each computation f_i guarantees ϵ_i-differential privacy, then $\mathbf{f}$ is $(\sum_{i=1}^{i=d} \epsilon_i)$-differentially private.*

3 Problem Formulation

In analogy to Definition 2, we formulate the problem of guaranteeing differential privacy to ILP in Definition 4.

Definition 4. *(Diff. Private ILP): An ILP algorithm f is ϵ-differentially private iff for any pair of neighboring databases [1] D_1 and D_2, for any $H \in \mathcal{L}_2$.*

$$\Pr(f(D_1) = H) \leq e^{\epsilon} \Pr(f(D_2) = H)$$

By Definition 4, the output hypothesis does not necessarily satisfy the three requirements stated in Definition 1. The reason is that by Definition 2, any differentially private algorithm must be randomized in nature where output range is defined as a property of the algorithm f regardless of the input database. As a result, Definition 4 defines a differentially private ILP algorithm over all the possible hypotheses instead of those "correct" ones.

4 Trade-off between Privacy and Utility

Although privacy is a very important problem in ILP, utility also matters; a trivial differentially private ILP algorithm can be generated by randomly outputting a hypothesis regardless of the database. Though private, this algorithm is useless in practice. Thus, we also need to quantify the utility of a hypothesis.

[1] In the differential privacy literature, databases are typically thought of as single tables. In a multi-relational setting, the proper definition of "neighboring" might change. For example, in a medical domain a neighboring database would remove one patient along with their respective prescriptions and diagnoses from other tables.

4.1 Our Utility Model

Our intuition for the utility model is to relax the requirements on hypotheses in Definition 1. In particular, we relax both the *validity* and *completeness* requirement, and introduce the notion of δ-*usefulness* ($0 \leq \delta \leq 1$).

Definition 5. *(δ-usefulness): A hypothesis H is* δ-useful *for the input database D iff $\exists D' \subseteq D$, and $\|D'\|/\|D\| \geq \delta$ such that*

1. *Approx. validity: $\forall h \in H$, $h \in M^+(D')$.*
2. *Approx. completeness: if a general clause g is true in $M^+(D')$, then $H \models g$.*
3. *Minimality: there is no subset of H which is validate and complete in D'.*

The notion of δ-usefulness quantifies the quality of a hypothesis in terms of the percentage of input database in which that hypothesis is correct. Furthermore, we define the quality of a differentially private ILP algorithm by its likelihood η to produce hypotheses of high quality. This is shown in Definition 6.

Definition 6. *((δ, η)-approximation): An ILP algorithm f is* (δ, η)-approximate *iff for any input database D,*

$$\Pr(f(D) \text{ is } \delta\text{-useful}) \geq 1 - \eta$$

Both δ and η are within the range of $(0, 1)$. Another way to understand the notion of (δ, η)-approximation is through the idea of PAC-learning [8] where the notion of "approximate correctness" is defined in terms of δ-usefulness. Next, we will quantify the trade-off between privacy and utility in ILP.

4.2 A Lower Bound on Privacy Parameter

Our techniques to prove the lower bound on the privacy parameter come from differentially private itemset mining [9]. Perhaps this is no surprise since both frequent itemset mining and association rule mining have been closely connected with the context of ILP [10] in which frequent itemset mining can be encoded as a ILP problem. We prove the lower bound on the privacy parameter ϵ if an ILP algorithm must be both ϵ-differentially private and (δ, η)-useful. This is shown in Theorem 3.

Theorem 3. *For any ILP algorithm that is both ϵ-differentially private and (δ, η)-useful,*

$$\epsilon \geq \frac{\ln(2^n(1-\eta))}{2((1-\delta)\|D\|+1)}$$

where n is the number of atoms in the language of hypotheses $\mathcal{L}_2$.

Proof. We model the language of the input database $\mathcal{L}_1$ as follows: each atom is taken from the set $I = \{a_1, \ldots, a_n\}$, and each individual's data is represented by a conjunctive clause of the atoms. We also model the language of the hypotheses $\mathcal{L}_2$ to be all the possible conjunctive clauses over the set of atoms I.

Suppose f is an ILP algorithm that is both ϵ-differentially private and (δ, η)-useful. To better understand our proof technique, we add another atom a_{n+1} to I, and then we construct an input database D of size m by including $\delta * m$ clauses of the form $h_1 = a_1 \wedge a_2 \wedge \ldots \wedge a_n \wedge a_{n+1}$. The rest are constructed as simply $h_2 = a_{n+1}$. Since the number of all the hypotheses including a_{n+1} is 2^n, there must exist a particular hypothesis h_3 such that $\Pr(f(B) = h_3) \leq 1/2^n$

Without loss of generality, let $h_3 = a_1 \wedge a_2 \wedge \ldots a_k \wedge a_{n+1}$. Then, we construct another database D' from D by replacing one clause of h_1 by h_3, and then every clause of h_2 by h_3. Thus, there is a total number of $\delta m - 1$ clauses of h_1 in B' and the rest of them being h_3. It is not hard to show that h_3 is the only δ-useful hypothesis in B: any subset of B of cardinality δm must contain at least one h_3, and thus, the δ-valid hypotheses are those that can be entailed by h_3. Hence, $\Pr(f(D') = h_3) \geq 1 - \eta$. Since D' and D differ by $2((1-\delta)m + 1)$ rows (one can think of this difference as the edit distance between two databases), by differential privacy,

$$1 - \eta \leq \frac{e^{\epsilon(2((1-\delta)m+1))}}{2^n}$$

Theorem 3 then follows.

The result of Theorem 3 is similar in flavor to [11], which proved that there is no differentially private algorithm that is able to answer $\mathrm{O}(n^2)$ count queries in a database of size n with reasonable accuracy. That is, if an ILP algorithm can be thought of as a sequence of count queries, and if the number of count queries exceeds a certain threshold, then the ILP algorithm cannot produce a result of high quality.

This is a discouraging result, which states that in general, it is very hard to simultaneously guarantee both differential privacy and a high utility requirement since $\|\mathcal{L}_2\|$ grows exponentially with the number of atoms. Theorem 3 suggests that in order to decrease the lower bound on the privacy parameter, we must either increase the size of the database $\|D\|$, or reduce the number of atoms in the hypotheses space $\mathcal{L}_2$. If a real world problem meets those two conditions, we might be able to get results of high quality while guaranteeing differential privacy. To verify this, we propose a differentially private ILP algorithm.

5 Differentially Private ILP Algorithm

In this section, we will first briefly describe a typical non-private ILP algorithm, inverse entailment as implemented in Aleph [12], and then show our revisions of the non-private algorithm to guarantee differential privacy. As we shift our focus to Aleph, we also extend from propositional logic to predicate calculus. In the rest of the paper, an atom is now an atomic formula of first-order logic, i.e., an n-ary predicate applied to a vector of n terms.

5.1 A Non-Private ILP Algorithm

The non-private ILP algorithm works as follows:

1. Select an example (selection): Select an example to be generalized.
2. Build most-specific-clause (saturation [13]): Construct the most specific clause that entails the example selected, and is within language restrictions provided. This is usually a definite clause with many literals, and is called the "bottom clause."
3. Search (reduction): Find a clause more general than the bottom clause. This is done by searching for some subset of the predicates in the bottom clause that has the "best" score.
4. Remove redundant (cover removal): The clause with the best score is added to the current hypothesis, and all examples made redundant are removed.

A careful analysis of the above steps shows that the selection and reduction steps directly utilize the input data while the saturation and cover removal steps depend on the output from the previous step. Thus, as discussed in literature [2], as long as we can guarantee the output from both selection and reduction is differentially private, then it is safe to utilize those output in subsequent computation. Hence, we only need to consider the problem of guaranteeing differential privacy for those two steps.

The input to the learning algorithm consists of a *target predicate*, which appears in the head of hypothesized clauses. The input database can be divided into two parts: the set of positive examples $E^+ \subseteq D$ which satisfy the target predicate, and the set of negative examples $E^- \subseteq D$ which do not. Furthermore, the bottom clause is normally expressed as the conjunctive form of the atoms, and thus we also use a "subset of the atoms" to denote the clause that is of the conjunctive form of the atoms in that subset.

5.2 A Differentially Private Selection Algorithm

The non-private selection algorithm is a sampling algorithm that randomly selects an individual's data to generalize. However, as discussed in [7], no sampling algorithm is differentially private. In this paper, we propose to use domain knowledge to overcome this obstacle. That is, we utilize the domain knowledge to generate a "fake" example. We want to emphasize that the domain information might come from external knowledge or previous interactions with the database. This information does not weaken the definition of differential privacy as stated in Definition 2, and we only utilize these previous known information to generate a fake example. In the worst case, this example can be expressed as the conjunction of all the predicates, which is considered as the public information. In that way, the new selection step does not rely on the input database, and thus, it is differentially private[2].

[2] An alternative is to relax the privacy definition from ϵ-differential privacy to (ϵ, δ)-differential privacy. In this context, δ refers to the probability that the algorithm violates the ϵ-differential privacy guarantee. We do not explore it here as it makes the already burdensome utility bounds much worse.

5.3 A Differentially Private Reduction Algorithm

The non-private reduction algorithm actually consists of two steps: 1) the heuristic method to search for a clause, which is a subset of predicates in the bottom clause, and 2) the scoring function to evaluate the quality of a clause. Although there are many different methods to implement the reduction algorithm [1], in this paper we follow the standard usage in Aleph in which the heuristic method is a top-down breadth-first search and the scoring function is coverage (the number of covered positive examples minus the number of covered negative examples). The search starts from the empty set and proceeds by the increasing order of the cardinality of the clauses until a satisfying clause is found. The pseudocode of the non-private reduction algorithm is shown in Algorithm 1.

Algorithm 1 NON-PRIVATE REDUCTION

Input: positive examples E^+; negative examples E^-; bottom clause H_b
Output:the best clause

1: k = number of atoms in H_b
2: $\mathcal{L}$ = the lattice on the subset of atoms in H_b
3: **for** $i = 1$ to k **do**
4: **for** each set $S \in \mathcal{L}$, $\|S\| = i$ **do**
5: P = the number of positive examples satisfying S
6: N = the number of negative examples satisfying S
7: $H^* = S$ if S has better coverage than the previously best clause
8: **end for**
9: **end for**
10: **return** H^*

A Naïve Differentially Private Algorithm We observe that the only part in Algorithm 1 that needs to query the input database is in the computation of P and N shown in line 5 and line 6, respectively. Therefore, as long as we can guarantee differential privacy in those two computations, then the reduction algorithm is differentially private. We do so by utilizing the geometric mechanism. Given a clause h, let q_h^+ and q_h^- be the queries that compute the number of positive examples and negative examples satisfying h, respectively. Then, as shown in Theorem 4, the sensitivity to evaluate a set of clauses is equal to the number of clauses in the set.

Theorem 4. *Given a set of clauses $H = \{h_1, h_2, \ldots, h_n\}$, and the corresponding evaluation queries $\mathbf{q} = \{q_{h_1}^+, q_{h_1}^-, \ldots, q_{h_n}^+, q_{h_n}^-\}$, the sensitivity of $\mathbf{q}$ is n.*

We show our differentially private reduction algorithm in Algorithm 2. By Theorem 4, Algorithm 2 is differentially private as shown in Theorem 5.

Theorem 5. *Algorithm 2 is ϵ-differentially private.*

Algorithm 2 Diff. Private Reduction

Input:positive examples E^+; negative examples E^-; bottom clause H_b; privacy parameter ϵ
Output:the best clause

k = number of atoms in H_b
$\mathcal{L}$ = build a lattice on the subset of atoms in H_b
for $i = 1$ to k **do**
    for each subset $h \in \mathcal{L}$, $\|S\| = i$ **do**
        $P' = q_h^+(E^+) + G(\epsilon/\|\mathcal{L}\|)$
        $N' = q_h^+(E^-) + G(\epsilon/\|\mathcal{L}\|)$
        $H^* = S$ if S has better coverage than previously best clause w.r.t. P',N'
    end for
end for
return H^*

The Relevance-Aware Differentially Private Reduction Algorithm We observe that Algorithm 2 has only considered the worst-case scenario in which the number of clauses to be evaluated is the whole lattice whereas in practice, the reduction algorithm seldom goes through every clause in the lattice. This occurs when criteria are set to specify unproductive clauses for pruning (preventing evaluation of supersets) or for stopping the algorithm. Thus, the number of clauses evaluated in practice is much less than that in the whole lattice, which means the scale of the noise added is larger than necessary. If the quality of a clause does not meet certain criterion, then there is no need to evaluate the subtree in the lattice rooted at that clause. This algorithm is shown in Algorithm 3.

Algorithm 3 Relevance-Aware Diff. Private Reduction

Input: positive examples E^+; negative examples E^-; bottom clause H_b; privacy parameter ϵ; levels ℓ
Output:the best clause

$\mathcal{L}$ = build a lattice on the subset of atoms in H_b
for $i = 0$ to ℓ **do**
    β = the number of clauses with k atoms existing in the lattice
    for each subset $h \in \mathcal{L}$, $\|S\| = i$ **do**
        $P' = q_h^+(E^+) + G(\frac{\epsilon}{\beta(\ell+1)})$
        $N' = q_h^+(E^-) + G(\frac{\epsilon}{\beta(\ell+1)})$
        $H^* = S$ if S has better coverage than the previously best clause
        if P' and N' does not meet the criterion **then**
            Delete the subtrees in the lattice rooted at S
        end if
    end for
end for
return H^*

We have also introduced another parameter ℓ in Algorithm 3 to specify the maximal cardinality of the desired clause, reducing the number of clauses to be evaluated. We prove Algorithm 3 is differentially private in Theorem 6.

Theorem 6. *Algorithm 3 is ϵ-differentially private*

5.4 Our Differentially Private ILP Algorithm

By using our differentially private selection algorithm and the relevance-aware differentially private reduction algorithm, we present our differentially private ILP algorithm in Algorithm 4. Since the output might consist of multiple clauses, we add the input parameter k which specifies the maximal number of clauses in the theory. We understand that this is not a usual setting for the usage of Aleph. Developing a differentially private algorithm that does not make use of k is an interesting challenge for future work.

Algorithm 4 Diff. Private ILP Algorithm

Input: positive examples E^+; negative examples E^-; privacy parameter ϵ; levels ℓ; rounds k
Output:the best theory

```
T = ∅
for i = 1 to k do
    H_b = Select a bottom clause in a differentially private way
    h = Relevance-Aware Diff. Private Reduction(E+, E−, ϵ/k, ℓ)
    Add h to T
    Remove redundant examples using h
end for
return T
```

By Theorem 2, Algorithm 4 is differentially private.

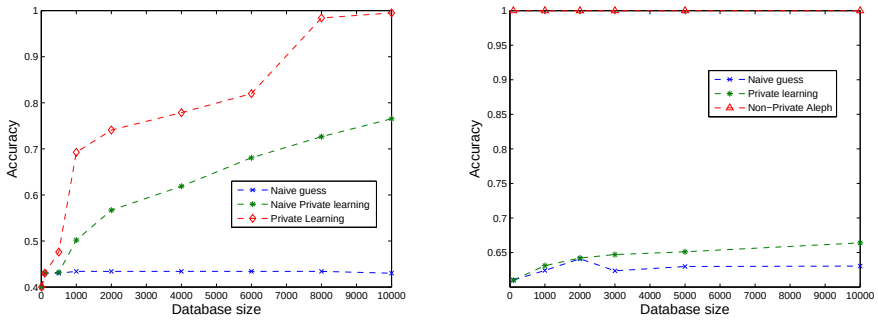
Theorem 7. *Algorithm 4 is ϵ-differentially private.*

6 Experiments

In our experiments, we run our differentially private algorithm on synthetic data generated by the train generator described in [13], in which the goal is to discriminate eastbound versus westbound trains based on the properties of their railcars. In all the experiments, we set the privacy parameter ϵ to be 1.0, and vary both the size of the data and the desired hypothesis to see how our algorithm performs. We measure the quality of our algorithm in terms of the accuracy of the output theory on a testing set. In all of our experiments, the naïve guess is the clause that assumes every train is eastbound.

In our first experiment, we consider a hypothesis of one clause. We vary the number of atoms in the clause as shown in Figure 1. As we can see in Figure 1a, when there are three atoms in the desired clause, our private learning

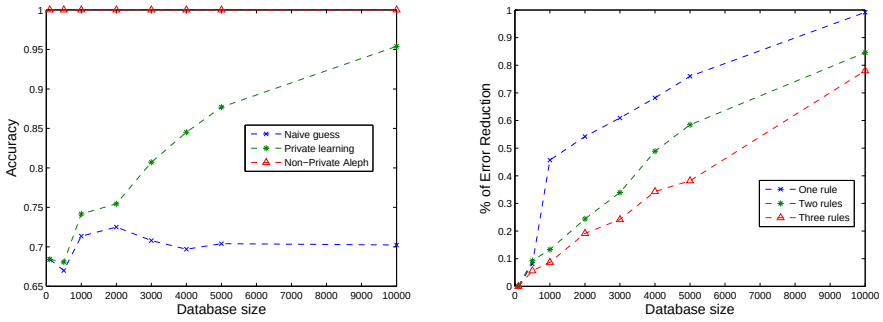
algorithm is able to learn the clause more accurately with more training data as discussed in Section 4. Furthermore, we also observe that our relevance-aware reduction algorithm shown in Algorithm 3 produces better results than the naïve reduction algorithm, demonstrating that reducing added noise by pruning low scoring clauses produces more accurate results. However, when increasing the number of atoms in the desired clause, the quality of our algorithm decreases. This is no surprise as the hypotheses space grows exponentially with the number of the atoms.



(a) Three Atoms (b) Six Atoms

Fig. 1: One Clause with Different Number of Atoms

We also investigate the effects on the number of clauses in a desired hypothesis, each of which consists of three atoms. In Figure 2a, our private learning algorithm produces high quality hypothesis with the growth in the size of the data, which is significantly better than the case of a single clause with six atoms. This is because the addition of a clause only increases the hypotheses space multiplicatively instead of exponentially. In both Figure 1 and Figure 2a we see that a large performance penalty is paid by the differentially private algorithms, as the non-private algorithm achieves perfect accuracy in all cases. Figure 2b shows the percentage of error reduction as more clauses need to be learned, showing the penalty due to the privacy budget being split among multiple clauses.



(a) Two Clauses (b) Multiple Clauses

Fig. 2: Multiple Clauses with the Same Number of Atoms

7 Related Work

The ILP has been well studied by the machine learning community, and see [1] [14] for a comprehensive survey on this field. ILP has been proved to be a great success in many applications ranging from natural language processing [15] to electronic health records processing [16] [17]. However, to the best of our knowledge, no previous work has addressed the privacy issues arising in the context of ILP, and our paper is the first one to discuss that problem.

The notion of differential privacy was proposed by Dwork et al. [2], and has received great attentions in both the machine learning community [3] [4] and the theory community [18] [11]. Dwork et al. [7] also propose the addition of Laplacian noise to guarantee differential privacy and [6] propose to add geometric noise to achieve the same goal. [11] proves that there is no differentially private algorithm that is able to answer $O(n^2)$ count queries in a database of size n with reasonable accuracy. Another way to guarantee differential privacy is to generate synthetic data in a differentially private way, and then run the non-private ILP algorithm on the private data [19]. However, the latest hardness result on differentially private data generation [20] [21] might prove this path elusive.

8 Conclusion

In this paper, we have proposed a differentially private ILP algorithm. We have precisely quantified the trade-off between privacy and utility in ILP, and our results indicate that in order to satisfy a non-trivial utility requirement, an ILP algorithm incurs a huge risk of privacy breach. However, we find that when limiting the hypotheses space and increasing the size of the input data, our algorithm is able to output a hypothesis with high quality on synthetic data set. To the best of our knowledge, ours is the first one to attack this problem. With the availability of privacy-sensitive data such as electronic health records, we hope more and more people begin to pay attention to the privacy issues arising in the context of ILP.

There are many potential opportunities for future work. One such direction would be to formalize the notion of differential privacy with first-order logic, and discuss the tradeoff between privacy and utility in that context. Furthermore, we have only considered ILP with definite clauses, and it would be interesting to expand our work to statistical relational learning [14]. Finally, since our algorithm requires one to limit the hypotheses space, it would also be interesting to investigate the feature selection problem in the context of differential privacy.

References

1. Muggleton, S., de Raedt, L.: Inductive logic programming: Theory and methods. Journal of Logic Programming (1994)
2. Dwork, C.: Differential privacy. In: ICALP. (2006)

3. Williams, O., McSherry, F.: Probabilistic Inference and Differential Privacy. In: Advances in Neural Information Processing Systems 23. (2010)
4. Rubinstein, B.I.P., Bartlett, P.L., Huang, L., Taft, N.: Learning in a large function space: Privacy-preserving mechanisms for svm learning. Journal of Privacy and Confidentiality (2012)
5. Chaudhuri, K., Monteleoni, C., Sarwate, A.D.: Differentially private empirical risk minimization. J. Mach. Learn. Res. (2011)
6. Ghosh, A., Roughgarden, T., Sundararajan, M.: Universally utility-maximizing privacy mechanisms. In: STOC. (2009)
7. Dwork, C., Mcsherry, F., Nissim, K., Smith, A.: Calibrating noise to sensitivity in private data analysis. In: TCS. (2006)
8. Valiant, L.G.: A theory of the learnable. Commun. ACM (1984)
9. Zeng, C., Naughton, J.F., Cai, J.Y.: On differentially private frequent itemset mining. Proc. VLDB Endow. (2012)
10. Dehaspe, L., Raedt, L.D.: Mining association rules in multiple relations. In: ILP. (1997)
11. Ullman, J.: Answering $n^{2+o(1)}$ counting queries with differential privacy is hard. STOC (2013)
12. Srinivasan, A.: Aleph manual
13. Muggleton, S.: Inverse entailment and progol. In: New Generation Computing
14. De Raedt, L.: Statistical relational learning: an inductive logic programming perspective. PKDD (2005)
15. Mooney, R.J.: Inductive logic programming for natural language processing. In: IN MUGGLETON, S. (ED.), INDUCTIVE LOGIC PROGRAMMING: SELECTED PAPERS FROM THE 6TH INTERNATIONAL WORKSHOP. (1997)
16. Page, D., Costa, V.S., Natarajan, S., Barnard, A., Peissig, P., Caldwell, M.: Identifying adverse drug events by relational learning. (2012)
17. Harpaz, R., Haerian, K., Chase, H.S., Friedman, C.: Mining electronic health records for adverse drug effects using regression based methods. In: Proceedings of the 1st ACM International Health Informatics Symposium. IHI '10 (2010)
18. Thaler, J., Ullman, J., Vadhan, S.P.: Faster algorithms for privately releasing marginals. In: ICALP. (2012)
19. Dwork, C.: Differential privacy: a survey of results. In: Proceedings of the 5th international conference on Theory and applications of models of computation. TAMC'08 (2008)
20. Ullman, J., Vadhan, S.: Pcps and the hardness of generating private synthetic data. In: Theory of Cryptography - 8th Theory of Cryptography Conference, TCC 2011. (2011)
21. Dwork, C., Naor, M., Reingold, O., Rothblum, G.N., Vadhan, S.: On the complexity of differentially private data release: efficient algorithms and hardness results. In: Proceedings of the 41st annual ACM symposium on Theory of computing. STOC '09 (2009)