Privacy Preserving Machine Learning: A Theoretically Sound Approach

Liwei Wang

Outline

Privacy Leakage Events

- **AOL** search data leak:

  New York Times journalist was able to identify users from the "anonymous" search data. The CTO of AOL resigned.

- **Netflix** $1 million learning for ratings contest:

  The "anonymous" users were cleverly identified by incorporating public information. Netflix was sued and had to cancel the second round of contest.

Studies of Data Privacy Outside of TCS

- Study of privacy in statistics.
- Study of Privacy in data mining.
- Limitations of these ad hoc methods: No mathematically rigorous definition of privacy.

Outline

Differential Privacy

- Idea of Differential Privacy: Almost nothing new can be learned from the database which contains an individual's information compared with that from the database without the individual's information.
- There is little risk for an individual to provide his/her data to the database.
- Database D: $D = \{\mathbf{x}_1, \ldots, \mathbf{x}_n\}$. Each $\mathbf{x}_i \in \mathcal{X}$ is the record of an individual.
- Two databases D and D' are said to be neighbors if $|D| = |D'|$, and they differ in exactly one individual record.

Differential Privacy

Definition: ϵ-differential privacy (Dwork et.al., '06)

A sanitizer $\mathcal{S}$ which is a randomized algorithm that maps input database into some range $\mathcal{R}$ is said to preserve ϵ-differential privacy, if for all pairs of neighbor databases D, D' and for every subset $A \subset \mathcal{R}$, it holds that

$$\mathbb{P}(\mathcal{S}(D) \in A) \leq \mathbb{P}(\mathcal{S}(D') \in A) \cdot e^{\epsilon}.$$

Definition: (ϵ, δ)-differential privacy

$$\mathbb{P}(\mathcal{S}(D) \in A) \leq \mathbb{P}(\mathcal{S}(D') \in A) \cdot e^{\epsilon} + \delta.$$

Privacy and Information

- Preserving differential privacy alone is easy: Simply output a fixed distribution regardless of the database D.
- Difficulty: How to output *useful* information of the database and preserve privacy simultaneously?
- Useful information: From a machine learning point of view, we only care about the information of population, i.e., distributional information rather than individual information.

Accuracy and Queries

- Measure usefulness of information: accuracy for queries.
- Linear queries: a query q_f is specified by a function $f : \mathcal{X} \to \mathbb{R}$,

$$q_f(D) := \frac{1}{|D|} \sum_{x \in D} f(x).$$

Definition: (α, β)-accuracy

Let Q be a set of queries. A sanitizer $\mathcal{S}$ is said to have (α, β)-accuracy for size n databases with respect to Q, if for every database D with $|D| = n$ the following holds

$$\mathbb{P}(\forall q \in Q, \quad |\mathcal{S}(D, q) - q(D)| \leq \alpha) \geq 1 - \beta,$$

where $\mathcal{S}(D, q)$ is the answer to q given by $\mathcal{S}$.

Outline

Laplace Mechanism

- Laplace mechanism: Give noisy answer to each query.

$$\mathcal{S}(D, q) = q(D) + \mathrm{Lap}(\sigma),$$

where $\mathrm{Lap}(\sigma)$ is the random variable distributed according to the Laplace distribution with parameter σ.

- Advantages: able to answer general linear queries; computationally efficient.

- Limitation: can answer at most $O(n^2)$ queries privately with non-trivial accuracy.

- In real applications there may be many users and each user may submit a set of queries. $O(n^2)$ is too restricted

Mechanisms Able to Answer $2^{n^{\Omega(1)}}$ Linear Queries

- BLR (Blum et.al., '08): Using the exponential mechanism (McSherry and Talwar '07) from a learning theoretic viewpoint.
- Median (Roth and Roughgarden '10): Private median algorithm.
- PMW (Hardt and Rothblum '10): Private Multiplicative Weight updating algorithm.
- Private Boosting (Dwork et.al, '10): Using a boosting approach.

Mechanisms Able to Answer $2^{n^{\Omega(1)}}$ Linear Queries

- Advantages: Able to answer a large number of general linear queries.
- Typical accuracy of these mechanisms are :

$$\alpha = O\left(\frac{\mathrm{polylog}(|\mathcal{X}|) \cdot \mathrm{polylog}(|Q|) \cdot \log(\frac{1}{\delta}) \cdot \log(\frac{1}{\beta})}{\sqrt{n} \cdot \epsilon}\right),$$

  where $\mathcal{S}$ is the data universe and Q is the query set.
- Disadvantages: computationally inefficient.

Classification of the Mechanisms

- BLR: output a synthetic database $\tilde{D}$.
- All other mechanisms: output answers to the queries, or output a summary of the database from which the answers can be computed.
- Outputting a synthetic database is highly preferred in applications. (Ad hoc methods only consider outputting synthetic DB.)
- Outputting a synthetic database is much more difficult than providing answers in terms of computational complexity.

Running Time of the Mechanisms

- PMW (the most efficient mechanism that outputs answers): the running time is linear in the size of the data universe $|\mathcal{X}|$ to answer one query.
- BLR: runs in time super-exponential in $|\mathcal{X}|$ and the number of queries $|Q|$.
- For these mechanisms, typically $\mathcal{X} = \{0, 1\}^d$, and $|\mathcal{X}|$ is much larger than $|D|$.

Outline

(Conditional) Hardness of Differential Privacy

- Hardness of privacy for general linear queries: Suppose $\mathcal{X} = \{0, 1\}^d$. There is no $\mathrm{poly}(n, d)$-time mechanism that can answer $n^{2+o(1)}$ general linear queries while preserving privacy and non-trivial accuracy (Ullman '12).

- Hardness for outputting synthetic database: there is no $\mathrm{poly}(n)$-time mechanism that can answer the set of queries specified by 2-way marginals (Ullman and Vadhan '11).

- 2-way marginals is a very small set of functions, only $\frac{d(d-1)}{2}$ queries! They can be efficiently answered by Laplace mechanism.

Summary of Private Mechanism for General Linear Queries

- Answering general linear queries is computationally intractable if $|\mathcal{X}|$ is large.
- Outputting a synthetic database is much more difficult than providing answers.

Outline

Differential Privacy for Restricted set of Queries

- If there is a class of queries which is rich enough to contain most queries used in practice and it allows one to develop efficient mechanisms, then the hardness results are not serious barriers for DP.

Differential Privacy for Restricted set of Queries

- k-way marginals: $\mathcal{X} = \{0, 1\}^d$. If k is a constant, then polynomial time mechanisms exist (Barak et.al. '07, Gupta et.al. '11, Thaler et.al. '12).
- All above mechanisms output answers of the queries.
- Rectangle queries: $\mathcal{X} = [-1, 1]^d$, where d is a constant. If $[-1, 1]^d$ is discretized to $\mathrm{poly}(n)$ bits of precision, then there is an efficient mechanism that outputs a synthetic DB.

Outline

(K, B)-smooth functions

- Let data universe $\mathcal{X} = [-1, 1]^d$, d is a constant.

Definition: (K, B)-smooth functions

Let $\mathbf{x} = (x_1, \ldots, x_d) \in [-1, 1]^d$. Let $\mathbf{k} = (k_1, \ldots, k_d)$ be a d-tuple of nonnegative integers. Define

$$D^{\mathbf{k}} := D_1^{k_1} \cdots D_d^{k_d} := \frac{\partial^{k_1}}{\partial x_1^{k_1}} \cdots \frac{\partial^{k_d}}{\partial x_d^{k_d}}.$$

Define the K-norm as

$$\|f\|_K := \sup_{|\mathbf{k}| \leq K} \sup_{\mathbf{x} \in [-1,1]^d} |D^{\mathbf{k}} f(\mathbf{x})|.$$

The set of (K, B)-smooth function is $C_B^K = \{f : \|f\|_K \leq B\}$

(K, B)-smooth Queries

- (K, B)-smooth queries:

$$Q_{C_B^K} := \left\{ q_f = \frac{1}{n} \sum_{\mathbf{x} \in D} f(\mathbf{x}) : \quad f \in C_B^K \right\}.$$

Fact: Gaussian kernel function is smooth

Let $f(\mathbf{x}) = \exp\left(-\frac{\|\mathbf{x}-\mathbf{x}_0\|^2}{2\sigma^2}\right)$, where $\mathbf{x}, \mathbf{x}_0 \in [-1, 1]^d$. Then for every $K \leq 2\sigma^2$, $\|f\|_K \leq 1$.

- Linear combination of Gaussian kernel functions are the most popular functions used in machine learning (SVM).

Outline

Main Results (Informal)

- We develop a polynomial time mechanism which outputs a synthetic database and is accurate for the set of all (K, B)-smooth queries for constant K, B.
- We develop a mechanism which provides accurate answers for the set of all (K, B)-smooth queries. The mechanism runs in sub-linear time (for answering a query) if the queries are of high order smoothness.

Main Results I

Theorem: Output synthetic DB in polynomial time

Let the query set be $Q_{C_B^K}$ defined on $[-1,1]^d$ (K, B, d are constants). Then there is a mechanism satisfying that for any $\epsilon > 0$, the following hold:

1) The mechanism preserves ϵ-differential privacy.

2) For every $\beta \geq \Omega(e^{-n^{\frac{1}{2d+K}}})$ the mechanism is (α, β)-accurate, where $\alpha = O(n^{-\frac{K}{2d+K}}/\epsilon)$.

3) The running time of the mechanism is $O(n^{\frac{3dK+5d}{4d+2K}})$.

4) The size of the output synthetic database is $O(n^{1+\frac{K+1}{2d+K}})$

Performance vs. Order of Smoothness

- Simplified results:

Order of smoothness	Accuracy α	Running time	Size of synthetic DB
$K = 1$	$O(n^{-\frac{1}{2d+1}})$	$O(n^2)$	$O(n^{1+\frac{2}{2d+1}})$
$K = 2d$	$O(n^{-\frac{1}{2}})$	$O(n^{\frac{3}{4}d+\frac{5}{8}})$	$O(n^{\frac{3}{2}+\frac{1}{4d}})$
$\frac{K}{d} = M \gg 1$	$O(n^{-(1-\frac{2}{M})})$	$O(n^{d(\frac{3}{2}-\frac{1}{2M})})$	$O(n^{2-\frac{1}{2M}})$

Comparison to Existing Mechanism

- Comparing to the performance of BLR, which is the only existing mechanism that can output synthetic DB.

Proposition: BLR for smooth queries

The accuracy guarantee of the BLR mechanism (with slight modification) on the set of K-smooth queries is

$$O\left(n^{-\frac{K}{d+3K}}\right),$$

which is at best $O(n^{-1/3})$ even if K/d is large. The running time for achieving such an accuracy is super-exponential in n.

Main Results II

- We give a mechanism consisting of two parts: one for outputting a synopsis of the database, the other for obtaining the answer of a smooth query from the synopsis.

Theorem: Output a synopsis

Let the query set be $Q_{C_B^K}$ defined on $[-1,1]^d$ (K, B, d are constants). Then there is a mechanism satisfying that for any $\epsilon > 0$, the following hold:

1) The mechanism is ϵ-differentially private.
2) For β exp. small the mechanism is $O(n^{-\frac{K}{2d+K}}/\epsilon)$-accurate.
3) The running time for $\mathcal{S}$ to output the summary is $O(n^{\frac{3d+K}{2d+K}})$.
4) The running time for $\mathcal{S}$ to answer a query is $\tilde{O}(n^{\frac{d+2+\frac{2d}{K}}{2d+K}})$

Performance vs. Order of Smoothness

- Simplified results:

Order of smoothness	Accuracy α	Time: output synopsis	Time: answer a query
$K = 1$	$O(n^{-\frac{1}{2d+1}})$	$O(n^{\frac{3}{2}})$	$\tilde{O}(n^{\frac{3}{2}+\frac{1}{4d+2}})$
$K = 2d$	$O(n^{-\frac{1}{2}})$	$O(n^{\frac{5}{4}})$	$\tilde{O}(n^{\frac{1}{4}+\frac{3}{4d}})$
$\frac{K}{d} = M \gg 1$	$O(n^{-(1-\frac{2}{M})})$	$O(n^{1+\frac{1}{M}})$	$\tilde{O}(n^{\frac{1}{M}(1+\frac{3}{d})})$

- Comparison: The accuracy guarantee of PMW for smooth queries is at best $n^{-\frac{1}{2}}$, and the running time for answering one query is $O(n^{\frac{d}{2}})$.

Ideas of the Algorithms (I)

- Smooth functions can be approximated by linear combination of a small set of basis functions $\phi_1(\cdot), \ldots, \phi_r(\cdot)$, in L_∞-norm.
- Special requirement for DP: the linear coefficients must be small. We require that the coefficients for all smooth functions are uniformly bounded by an absolute constant.
- If such small-coefficient approximation is valid, we can treat the basis functions as basis queries, and give noisy answers to these basis queries.
- For each $f \in C_B^K$, if we know linear coefficients $f \approx \sum c_i \phi_i$,

Ideas of the Algorithms (II)

- Q1: What basis satisfy the small-coefficient requirement?
- First idea: we transform the smooth functions and use trigonometric polynomials as the basis. We show such basis functions have the small-coefficient property.
- How to compute the linear coefficients?
- The coefficients can be computed. In our second mechanism, we develop complicated numerical integration methods to approximately compute the coefficients to a precision so that the accuracy of the mechanism is guaranteed, and utilize the smoothness property to significantly reduce running time.

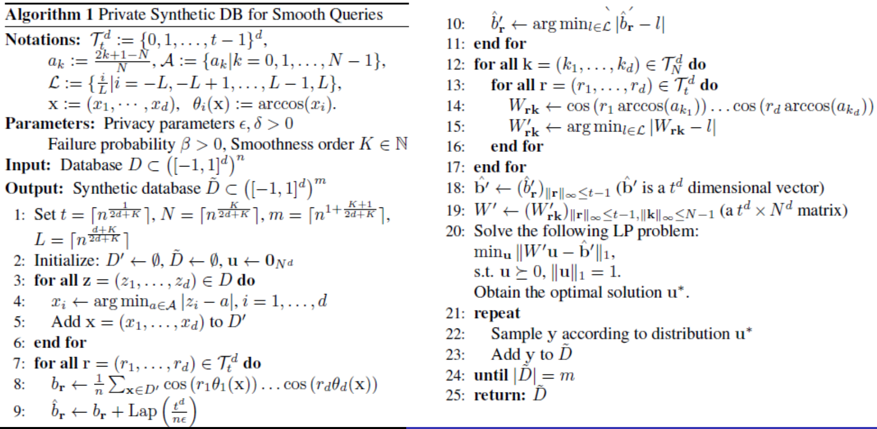
Ideas of the Algorithms (III)

- Q2: Do we need to know the coefficients?
- Second idea: We do not need to know the coefficients at all. We only need to know there *exist* such small coefficients.
- Reason: Our goal is to output a synthetic database $\tilde{D}$. If the answers of all the basis queries on D and $\tilde{D}$ are close, then their answers to all smooth queries are approximately the same since all the linear coefficients are small.

Ideas of the Algorithms (IV)

- We try to find $\tilde{D}$ so that: 1) the answers of $\tilde{D}$ to all basis queries are close to the noisy answers of D; 2) $\tilde{D}$ does not get any other information from D. These guarantee privacy and accuracy.
- We first find a distribution over $[-1, 1]^d$ which satisfies the above two requirements. This can be formulated as a linear programming problem. (We need to discretize $[-1, 1]^d$ to a certain precision and obtain a discrete distribution.)
- Finally, sampling sufficiently many samples from the distribution forms the synthetic database.

The Mechanism that outputs synthetic DB



Algorithm 1 Private Synthetic DB for Smooth Queries

```
Notations: $\mathcal{T}_t^d := \{0, 1, \ldots, t-1\}^d$,
    $a_k := \frac{2k+1-N}{N}$, $\mathcal{A} := \{a_k | k = 0, 1, \ldots, N-1\}$,
    $\mathcal{L} := \{\frac{i}{L} | i = -L, -L+1, \ldots, L-1, L\}$,
    $\mathbf{x} := (x_1, \cdots, x_d)$,  $\theta_i(\mathbf{x}) := \arccos(x_i)$.
Parameters: Privacy parameters $\epsilon, \delta > 0$
    Failure probability $\beta > 0$, Smoothness order $K \in \mathbb{N}$
Input: Database $D \subset \left([-1,1]^d\right)^n$
Output: Synthetic database $\tilde{D} \subset \left([-1,1]^d\right)^m$
 1: Set $t = \lceil n^{\frac{1}{2d+K}} \rceil$, $N = \lceil n^{\frac{K}{2d+K}} \rceil$, $m = \lceil n^{1+\frac{K+1}{2d+K}} \rceil$,
    $L = \lceil n^{\frac{d+K}{2d+K}} \rceil$
 2: Initialize: $D' \leftarrow \emptyset$, $\tilde{D} \leftarrow \emptyset$, $\mathbf{u} \leftarrow \mathbf{0}_{N^d}$
 3: for all $\mathbf{z} = (z_1, \ldots, z_d) \in D$ do
 4:     $x_i \leftarrow \arg\min_{a \in \mathcal{A}} |z_i - a|$, $i = 1, \ldots, d$
 5:     Add $\mathbf{x} = (x_1, \ldots, x_d)$ to $D'$
 6: end for
 7: for all $\mathbf{r} = (r_1, \ldots, r_d) \in \mathcal{T}_t^d$ do
 8:     $b_{\mathbf{r}} \leftarrow \frac{1}{n} \sum_{\mathbf{x} \in D'} \cos(r_1 \theta_1(\mathbf{x})) \ldots \cos(r_d \theta_d(\mathbf{x}))$
 9:     $\hat{b}_{\mathbf{r}} \leftarrow b_{\mathbf{r}} + \mathrm{Lap}\left(\frac{t^d}{n\epsilon}\right)$
10:     $\hat{b}'_{\mathbf{r}} \leftarrow \arg\min_{l \in \mathcal{L}} |\hat{b}_{\mathbf{r}} - l|$
11: end for
12: for all $\mathbf{k} = (k_1, \ldots, k_d) \in \mathcal{T}_N^d$ do
13:     for all $\mathbf{r} = (r_1, \ldots, r_d) \in \mathcal{T}_t^d$ do
14:         $W_{\mathbf{rk}} \leftarrow \cos(r_1 \arccos(a_{k_1})) \ldots \cos(r_d \arccos(a_{k_d}))$
15:         $W'_{\mathbf{rk}} \leftarrow \arg\min_{l \in \mathcal{L}} |W_{\mathbf{rk}} - l|$
16:     end for
17: end for
18: $\hat{\mathbf{b}}' \leftarrow (\hat{b}'_{\mathbf{r}})_{\|\mathbf{r}\|_\infty \le t-1}$ ($\hat{\mathbf{b}}'$ is a $t^d$ dimensional vector)
19: $W' \leftarrow (W'_{\mathbf{rk}})_{\|\mathbf{r}\|_\infty \le t-1, \|\mathbf{k}\|_\infty \le N-1}$ (a $t^d \times N^d$ matrix)
20: Solve the following LP problem:
    $\min_{\mathbf{u}} \|W'\mathbf{u} - \hat{\mathbf{b}}'\|_1$,
    s.t. $\mathbf{u} \succeq 0$, $\|\mathbf{u}\|_1 = 1$.
    Obtain the optimal solution $\mathbf{u}^*$.
21: repeat
22:     Sample $\mathbf{y}$ according to distribution $\mathbf{u}^*$
23:     Add $\mathbf{y}$ to $\tilde{D}$
24: until $|\tilde{D}| = m$
25: return: $\tilde{D}$
```

Remarks

- The running time of the mechanism is dominated by the LP step. The LP problem has $n^{\frac{Kd}{2d+K}}$ variables, $n^{\frac{d}{2d+K}}$ constraints. (#variables $\gg$ #constraints)
- The number of variables is the number of grids obtained by discretizing $[-1, 1]^d$; The number of constrains is the number of basis functions.

Outline

Practical Variant of the Algorithm

- Randomly choose a subset of size C from the $n^{\frac{Kd}{2d+K}}$ grids.
- Randomly choose a subset of size R from the $n^{\frac{d}{2d+K}}$ trigonometric basis, favoring those with low degrees.
- Remark: This variant still rigorously guarantee privacy. But there is no theoretical guarantee of accuracy. (Or there is but we do not know?)

Settings of the Experiments

- Use linear combination of Gaussian Kernel function as the smooth queries.
- Randomly choose 10^4 queries and calculate the worst case error. Both absolute and relative errors are considered.
- Use CPLEX to solve LP.
- Conduct experiments on a workstation with 2 Intel Xeon X5650 processors of 2.67GHz and 32GB RAM.

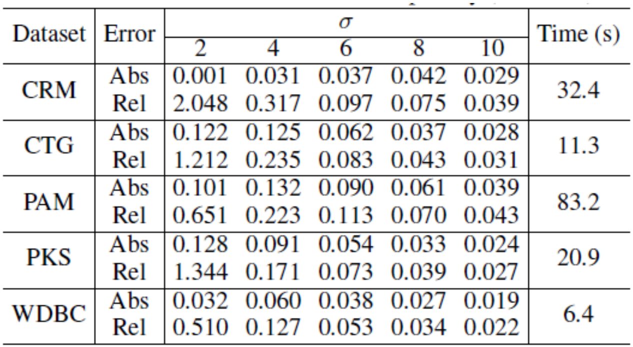
Databases Containing Sensitive Information

- Description of databases

Dataset	Size (n)	# Attributes (d)
CRM	1993	100
CTG	2126	20
PAM	20000	40
PKS	5875	20
WDBC	569	30

Experimental Results I

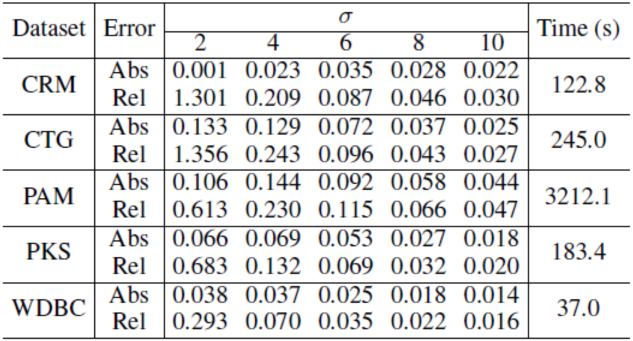
- Worst case error of the ϵ-differentially private mechanism ($C = 10^4$)



Dataset	Error	σ					Time (s)
		2	4	6	8	10	
CRM	Abs	0.001	0.031	0.037	0.042	0.029	32.4
	Rel	2.048	0.317	0.097	0.075	0.039	
CTG	Abs	0.122	0.125	0.062	0.037	0.028	11.3
	Rel	1.212	0.235	0.083	0.043	0.031	
PAM	Abs	0.101	0.132	0.090	0.061	0.039	83.2
	Rel	0.651	0.223	0.113	0.070	0.043	
PKS	Abs	0.128	0.091	0.054	0.033	0.024	20.9
	Rel	1.344	0.171	0.073	0.039	0.027	
WDBC	Abs	0.032	0.060	0.038	0.027	0.019	6.4
	Rel	0.510	0.127	0.053	0.034	0.022	

Experimental Results II

- $C = 5 \cdot 10^4$: More grids leads to better accuracy, but significantly more running time.



Dataset	Error	σ					Time (s)
		2	4	6	8	10	
CRM	Abs	0.001	0.023	0.035	0.028	0.022	122.8
	Rel	1.301	0.209	0.087	0.046	0.030	
CTG	Abs	0.133	0.129	0.072	0.037	0.025	245.0
	Rel	1.356	0.243	0.096	0.043	0.027	
PAM	Abs	0.106	0.144	0.092	0.058	0.044	3212.1
	Rel	0.613	0.230	0.115	0.066	0.047	
PKS	Abs	0.066	0.069	0.053	0.027	0.018	183.4
	Rel	0.683	0.132	0.069	0.032	0.020	
WDBC	Abs	0.038	0.037	0.025	0.018	0.014	37.0
	Rel	0.293	0.070	0.035	0.022	0.016	

Conclusion

- We develop a polynomial time mechanism which outputs a synthetic database and is accurate for the set of smooth queries.
- We develop a mechanism which provides accurate answers for the set of all smooth queries. The mechanism runs in sub-linear time if the queries are of high order smoothness.
- These mechanisms can be easily generalized to preserve (ϵ, δ)-dp with slightly better accuracy.
- The practical version of the mechanism works well on real databases.

Future Works

- In almost all real databases, each individual record contains both categorical and continuous attributes. How to design a mechanism that can deal with both types of attributes?
- For data with categorical attributes ($\mathcal{X} = \{0, 1\}^d$), there are other important classes of query functions worth analyzing. For example, does the set of submodular functions allow efficient mechanisms?

References

- Wang et al, Efficient Algorithm for Privately Releasing Smooth Queries, *NIPS*, 2013.