The optimization of the Range-Count Queries in Differential Privacy

Lei Qian, Tao Song, Alei Liang
Swarm Intelligence Laboratory, Shanghai Jiao Tong University
Email: {bgw991111, songt333, liangalei}@sjtu.edu.cn

***Abstract*—Privacy-preserving data publishing has been widely explored in academia recently. The state-of-the-art goal for data privacy-preserving is *differential privacy*, which offers a strong degree of privacy protection against adversaries with arbitrary background knowledge. However, along with a wide query scope in the non-interactive model like DiffGen, the accumulation of noise in the query answers can effect the usability of the released data. In this paper, we present *Consistent DiffGen*(*CDiffGen*), a non-interactive differentially-private algorithm that is aimed at range-count queries and optimises the DiffGen module with the consistency constraints among data attributes. We experimentally evaluate *CDiffGen* on real dataset and the result performs the effectiveness and the improvement of our solution in range-count query tasks.**

***Index Terms*—Differential privacy; range-count query; non-interactive model; consistency; sensitivity.**

I. Introduction

The private information within the dataset is always a major concern for the owner of a database. Traditional privacy-preserving methods based k-anonymization[1] can protect privacy,such as l-diversity[2] and (α,k)-anonymity[3]. Once adversaries obtain the background information, they can take new attack approaches, for instance, Composition Attack[4] and Foreground Knowledge Attack[5]. In such situations, the traditional privacy-preserving methods are not so safe.

Differential privacy is a rigorous privacy model that provides strong privacy guarantees, in spite of the background knowledge and attack approached that the adversaries have. A differentially-private mechanism perturbs the raw statistics D with a randomised algorithm Ψ, and the output of algorithm $O = \Psi(D)$ remains roughly the same even if any single tuple in raw dataset is arbitrary modified. Thus, the differentially-private mechanism guarantees that all outputs are insensitive to any individual's data, and the adversaries can't infer any tuple from the released data.

DiffGen[6] is a differentially-private anonymization algorithm, so it makes a anonymous rules convert the original attributes to new attributes. But DiffGen's direct way for injecting noise has a poor effect on range-count queries owing to the accumulation of noise. The noise in the query answers can be proportional to the number of entries in the data[7], which renders the results useless for subsequent analysis. The problem is existed in DiffGen as well as Dwork et al.'s[8] meets in noisy frequency matrix.

Example 1. *$TableI(a)$ shows a real estate statistics for eight people of different ages, and $TableI(b)$ shows the relevant frequency matrix. $YesCount$ and $NoCount$ respectively represent the count of having house and no.*

TABLE I. Records and Frequency Matrix

(a) Real estate records

Age	$HasHouse(s)$
<20	No
20-30	Yes
30-40	No
20-30	No
40-50	Yes
30-40	Yes
<20	No
30-40	No

(b) Frequency Matrix

Age	$YesCount$	$NoCount$
<20	0	2
20-30	1	1
30-40	1	2
40-50	1	0

With a differentially-private mechanism to protect the raw statistics, Dwork et al.'s injects independent noise with $\Theta(1)$ variance into every entry and then gets noisy frequency matrix. The *sensitivity* of this query set is 1. When a query involves to several interlinking entries, the noise-accumulation will happen. In Example 1, if an analyst requests a m-entry query under the same Age, the noisy answer will be Θ(m) variance(the m entries are related with attribute Age). For example, a analyst sends a query in 20-40 years old, the owner of the database returns the noisy answer by adding up the entries involved in Age of {20-30} and {30-40}. Obviously, the more entries the query involved in, the more noise-accumulation happens. This would make the answer inaccurate or even useless.

In this paper, we propose a powerful solution, dubbed *Consistent DiffGen*(*CDiffGen*) in term of range-count queries. It makes use of the consistency constraints among data attributes to optimize the noise distribution, which is based on the premise of meeting differential privacy. The contributions of this paper are summarized as follow:

1) As the challenge Dwork et al.'s meets, we reveal the similar problems existing in DiffGen in the matter of

range-count queries, namely the correlation between noise variance and dimension of query scope.

2) To optimize DiffGen, we summarize the consistency property among the data attributes in DiffGen model. Based on the auxiliary tree structure, we can take advantage of consistency as bounds to remove redundant noise between parent-child node pairs. After redefining the sensitivity, *CDiffGen* still satisfies ε-differential privacy on the condition of a noise budget ε and takes simple data-released pattern as DiffGen.

The rest of the paper is organized as follows. Section II overviews ε-differential privacy and states the problem with quantization perspective. Section III describes the existing problems and CDiffGen algorithm. The experimental evaluation is illustrate in Section IV. Section V concludes the paper.

II. Preliminaries

A. Differential Privacy

In query model, let D be a dataset and $Q = \{q_1, q_2, ..., q_m\}$ represents a query sequence that consists of m query requests. We expect the answers of query sequence are insensitive to small changes in the D. Let D' be a dataset that differ from D by at most one record. In differential privacy, D and D' will be modify by a randomized algorithm, and the difference of output distribution between the two datasets is don't reveal any one particular record.

Definition II.1. (ε-Differential Privacy[8]) A randomized algorithm ∂ satisfy differential privacy, if for datasets D and D' differing on at most one record($|D \triangle D'| \leqslant 1$), and for any possible output O of ∂, we have

$$Pr[\partial(D) = O] \leqslant e^{\varepsilon} \cdot Pr[\partial(D') = O] \tag{1}$$

where the Pr represents the probability of event happen.

To achieve differential privacy, there are many approaches. In this paper, we are interested in two widely used approaches, namely the $Laplace\ mechanism$[8] and $exponential\ mechanism$[9]. Both of them are based of the concept of $sensitivity$, which represents the maximum change of a function output when modifying one tuple in dataset.

Definition II.2. (Sensitivity[8]) Datasets D and D' are differing on at most one record, and let $\mathbb{F}$ to be a function that map D or D' to a d-size vector of real numbers: $D \rightarrow \mathrm{R}^d$. The sensitivity of $\mathbb{F}$ denoted as $S(\mathbb{F})$ is defined as

$$S(\mathbb{F}) = \max_{D,D'} \|\mathbb{F}(D) - \mathbb{F}(D')\|_1 \tag{2}$$

where the $\|\cdot\|$ denotes the L_1 norm, and $|D \triangle D'| \leqslant 1$.

The exponential mechanism is suitable for categorical and numeric attribute. Given a specified $score\ function$, the exponential mechanism selects a attribute with the probability that is proportional to a score.

With function $\mathbb{F}$, Laplace mechanism takes as input a dataset and output a set of numeric values . Laplace mechanism realizes differential privacy by sampling noise by Laplace distribution and injecting noise to query counts.

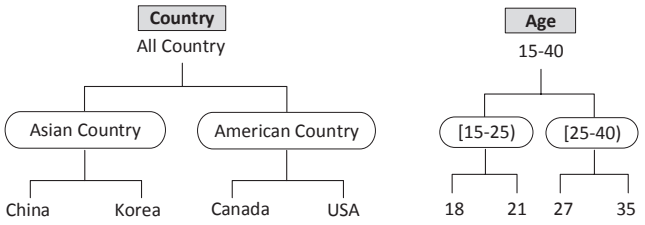


Fig. 1: Taxonomy tree

Theorem II.1. *(*Laplace Mechanism*[8]) Let $\mathbb{F}$ to be a function: $D \rightarrow \mathrm{R}^d$, and $\widetilde{\mathbb{F}}$ is the perturbed output. The computation*

$$\widetilde{\mathbb{F}}(D) = \mathbb{F}(D) + (Laplace(S(\mathbb{F})/\varepsilon)^d \tag{3}$$

provides ε-differential privacy.

The magnitude of the noise depends on $S(\mathbb{F})/\varepsilon$. For example, in frequency matrix the $\mathbb{F}(D)$ is a counting function on dataset D, that is $\mathbb{F}(D) = |D|$, and than the $S(\mathbb{F})$ equals 1. In this case, the noise added depends on $Laplace(1/\varepsilon)$ to satisfy ε-differential privacy.

B. Problem Statement

Suppose a raw dataset D consists of d attributes, denoted as $A_1, A_2, ..., A_d$ and let n be the number of non-zero entries. The size of dimension for dataset is expressed as *Domain Size*(M). M is the product of cardinalities of the attributes[10] and the size of contingency table. It is obvious that the $M = \prod_{i=1}^{d} |A_i| \gg n$ easily. For example, a dataset with 10 attributes and each of cardinality is 10. Apparently, $m = 10^{10}$ in spite of only ten attributes, and its contingency table would be huge. The scope of a query includes $A_k, A_{k+1}, ..., A_j$, $k \leqslant j$, the orders of magnitude that entry involved in would be $\prod_{i \in (1,d)-(k,j)} |A_i|$. It is seen that the table-lookup process becomes difficult and time-consuming. Once a differential privacy algorithm injects noise with $\Theta(1)$ variance into every entry, the amount of noise would make up the majority of answer because of the accumulation of noisy counts.

Given a dataset D, our objective is to make *Consistent DiffGen* satisfy ε-differential privacy with a given private budget ε, and optimizing the distribution of noise for a more accurate answer.

III. Consistent DiffGen Algorithm

In this section, we first show the anonymization procedure with *Taxonomy Tree* and *Gereralization* process. And then, we make an overview of DiffGen and analyze its request-response. As the challenge Dwork et al.'s meets, we reveal that DiffGen encounters the similar problems in the matter of range-count queries. Finally, by the consistent property in *Taxonomy Tree*, we present the *Consistent DiffGen* algorithm and the implementation details.

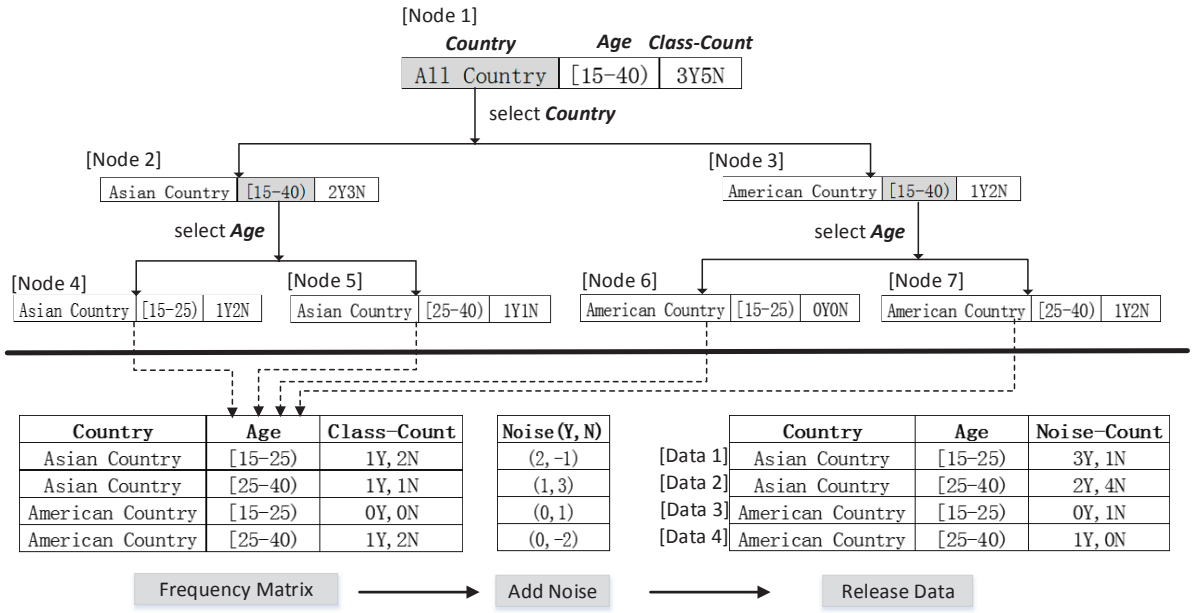


Fig. 2: DiffGen tree and attribute partitions

TABLE II. RAW DATA AND GENERALIZED DATA

(a) Raw data table

Country	*Age*	*Class*
China	18	N
Korea	21	Y
Canada	27	N
USA	35	N
USA	29	Y
China	39	Y
Korea	22	N
China	28	N

(b) Generalized data table

Country	*Age*	*ClassCount*
Asian Country	[15-25)	3
Asian Country	[25-40)	2
American Country	[15-25)	0
American Country	[25-40)	3

A. Taxonomy Tree

DiffGen[6] is a non-interactive differential privacy model based anonymization algorithm and defines a set of *Taxonomy Tree*(TaxoTree) as anonymization guideline. Each attribute corresponds to a TaxoTree, and every node in TaxoTree is a anonymous attribute, dubbed $Attribute\ Partition$(AP).

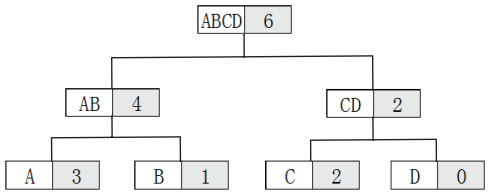
Example 2. *In a company, there are 8 interviewees as shown* Table II(a). *There are attributes* Country *and* Age*, and* Class *is* Y *and* N*, which respectively means weather the interviewee is hired or not.*

As for Example 2, Figure 1 shows the taxonomy trees for attribute *Country* and *Age*. In a TaxoTree, none-leaf nodes are anonymous attributes that are abstracted from lower nodes. Starting from the root node by a top-down view, the DiffGen makes a *Specialization* rules to guide how to generate children AP from parent AP. Conversely, the *Generalization* process guides the children AP how to be merged into a upper AP. For a numerical attribute, *Generalizaiton* is combining interval, and for a categorical attribute, it generates a more general concept. Table II(b) presents the statistical result for raw data based on TaxoTree. $ClassCount$ is a raw sum of Y and N under corresponding attribute partition.

B. DiffGen And Existing Problems

With the TaxoTree as guidelines, the raw data is encapsulated within root attribute partition firstly, and then partitioned by layers until a lowest partition level defined in advance, such as the bubble attribute level. We illustrate the whole procedure of DiffGen and the following paragraph with Figure 2.

A tree structure that consists of $[Node_{1-7}]$ is presented as *DiffGen Tree*(DT). It is located in the upper half of Figure 2, which is divided by the central heavy line. After building TaxoTree, all raw data are encapsulated within the root node $[Node_1]$. In the DiffGen algorithm, each iteration selects a candidate by $Exponential\ Mechanism$ to 'cut' the current node to generate children nodes, which is the mentioned $Specialization$ process. For example, in node $[Node_1]$, DiffGen selects $Country$ as candidate and then partitioning into $[Node_2]$ and $[Node_3]$ by $Country$'s TaxoTree. And so on, the raw data is scattered on leaf nodes $[Node_{4-7}]$ and obtaining raw frequency matrix. After injecting Laplace noise



(a) Tree structure of string

Answer: {C_A, C_B, C_C, C_D, C_{AB}, C_{CD}, C_{ABCD}}

True Answer: {3, 1, 2, 0, 4, 2, 6}

Noise Answer: {2, 3, 0, 3, 6, 0, 7}

(b) Query answer

Fig. 3: Example for consistency

to frequency matrix entries, DiffGen releases differentially-private dataset eventually.

We make reverse thinking to the process. Every non-leaf node's $Class\text{-}Count$ value can be got with the combination of its leaf nodes, which looks like to search the leaf node set that can be 'covered' by the current node. Expanding each entry in $Released\ Dataset$, we can all find a path in DT that is mapped to the entry from a certain node to a leaf node. Therefore, in terms of injecting noise, DiffGen takes the same method as Dwork et al.'s[8]— directly injecting noise to each entry with $\Theta(1)$ variance in frequency matrix.

It seems to be acceptable for the maximum accumulation process involved in 4 leaf nodes in Figure 2. But when a dataset has a huge domain size, $M = \prod_{i=1}^{d} |A_i|$, there are more attributes(the bigger d) and each attribute has more values(the bigger $|A_i|$). The complicated accumulation process will drastically reduce the accuracy with M increasing by times.

C. Consistent Property

Figure 3(a) indicates a simple DT containing character 'A', 'B', 'C' and 'D', and leaf nodes can be grouped into a inner node easily by *Generaliztion* process(except for 'string AD' and 'BC'). A DT node is expressed as a square frame— the left is character and the right part is its counts. Every leaf node maps to a character and the non-leaf node denotes a string. We express the count of string or character X as C_X and noisy count as $\tilde{C}_X$. With the consistent relation, we get equations:

$$C_{AB} = C_A + C_B \tag{1}$$

$$C_{CD} = C_C + C_D \tag{2}$$

$$C_{ABCD} = C_{AB} + C_{CD} \tag{3}$$

These equations on consistent constraint gives the arithmetic bounds to noisy count operation. For example, the summation of noisy counts $\tilde{C}_A$ and $\tilde{C}_B$ should be even more rounded to $\tilde{C}_{AB}$. Therefore, the changes to the noise-injection strategy and the release form are likely to get a more accurate answer.

In the DiffGen, the release form $\mathcal{F}$ is represented as $\mathcal{F} = \{\tilde{C}_A,\tilde{C}_B,\tilde{C}_C,\tilde{C}_D\}$. In the CDiffGen, we adopt a new release form, which is injecting noise to every node appearing at DT. By this time, the the released data denotes as $\mathcal{F}'$, that is $\mathcal{F}' = \{\tilde{C}_A,\tilde{C}_B,\tilde{C}_C,\tilde{C}_D,\tilde{C}_{AB},\tilde{C}_{CD},\tilde{C}_{ABCD}\}$, as shown in Figure 3(b). Contrasting with $\mathcal{F}$, $\mathcal{F}'$ can minimize unnecessary accumulation by selecting a suitable parent node set that covers the targeted leaf nodes as many as possible.

Example 3. *In a case of a query for string 'ABCD', there are multiple choices in $\mathcal{F}'$, namely $\{\tilde{C}_{ABCD}\}$,$\{\tilde{C}_{AB}$+$\tilde{C}_{CD}\}$ or $\{\tilde{C}_A$+C_B+$\tilde{C}_C$+$\tilde{C}_D\}$. The owner of dataset selects $\{\tilde{C}_{ABCD}\}$ clearly. If the query is 'BCD', the $\{\tilde{C}_B$+$\tilde{C}_{CD}\}$ is optimal selection.*

This is a $Minimum\ Set\ Covering$ problem in tree structure, as shown in $Algorithm\ 1$. Given a leaf-node set in a tree, our priority is seeking the minimal amount of parent nodes that can cover leaf-node set thoroughly.

Algorithm 1 Minimun set covering algorithm in DiffGen Tree

Input: DiffGen Tree $DTtree$, Leaf Node Set $\mathring{A}$
Output: Node Cover Set Υ Initialize Υ, $\Upsilon \leftarrow \emptyset$

```
for each leaf node x∈DTtree by postorder do
    x's leaf nodes set is å
    while å⊆ Å do
        á ← å
        x ← x's parent node
    end while
    Υ ← Υ ⋃ á
    if Υ is equal to Å then
        return Υ
    end if
end for
```

In new release form $\mathcal{F}'$, we change the noise distribution that injecting noise to every node appearing at DT. The $Sensitivity$ need to be redefine naturally on account of the new noise-injection strategy. It is no longer independent in parent-child node pairs based on DT structure. The $Sensitivity$ is equal to the height of DT[11]. The height of the DT is ℓ and the total number of nodes is t.

Theorem III.1. *The sensitivity of DT is ℓ.*

By $Theorem$ II.1, the following algorithm provides ε-differential privacy:

$$\widetilde{\mathbb{F}}(D) = \mathbb{F}(D) + (Laplace(\ell/\varepsilon)^t$$

Apparently, with the tree height ℓ increasing, the rise in sensitivity is likely to make the answer for single point query inaccurate slightly, but the promotion to range-count queries will significant fill this gap.

Inspired by Hay[12], the consistent constraint can be used to adjust noise distribution and obeying differential privacy simultaneously. And Hay et al.'s only supports full κ-ary tree model($\kappa \in \mathbb{N}+$), but we expand it to arbitrary tree structure.

D. Implementation

By the aid of DiffGen algorithm, we get a intact DT, dubbed $DTree$. A node x in DT with noisy count denoted as $\tilde{x}$, and after adjusting noise distribution it becomes $\vec{x}$. With x to be root node in a subtree $Xtree$, the number of $Xtree$ is $N_{leaf}(x)$ and the total number of nodes is $N_{total}(x)$. x's parent node is $P(x)$ and the set of its children is $Child(x)$. The $Algorithm$ 2 shows the procedure of how to build *Consistent DiffGen Tree*.

Algorithm 2 Consistent DiffGen to Adjust Noise Distribution

Input: DiffGen Tree $DTree$,
Output: Consistent DiffGen Tree $CDTree$

for each node $x \in DTree$ by postorder traversal **do**
  if x is leaf node **then**
    $\vec{x} \leftarrow \tilde{x}$
  else
    $\vec{x} \leftarrow \frac{N_{leaf}(x)*\tilde{x}+(N_{total}(x)-N_{leaf}(x))*\sum_{j\in Child(x)}\vec{j}}{N_{total}(x)}$
  end if
end for
for each node $x \in DTree$ by preorder traversal **do**
  if x is root node **then**
    $\vec{x} \leftarrow \vec{x}$
  else
    $ChildNum = \sum_{j\in Child(x)} 1$
    $\vec{x} \leftarrow \vec{x} + \frac{\overrightarrow{P(x)}-\sum_{j\in Child(P(x))}\vec{j}}{ChildNum}$
  end if
end for
return new $DTree$ as $CDTree$

To build CDiffGen, we traverse DT for two times. Firstly in a the bottom-up order, we scatter the noise over all nodes in the tree. And then from top-down, we adjust the noise distribution for purpose of obtaining consistency property. After adjusting operation, instead of releasing all tree nodes in DT, we can only make the combination of leaf nodes answer to all kinds of requests, like that in DiffGen. Because we recover the consistent property on DT and we can get arbitrary inner node by combining the several leaf nodes.

IV. Experimental Evaluation

In this section, we mainly evaluate the CDiffGen and DiffGen with real dataset in term of classification accuracy rate, and explore the experiment phenomena.

A. Experimental Settings

We employ the real Adult[13] dataset that contains census data. And we define the same TaxoTree adopted in the training phase and testing phase, which makes the DiffGen and CDiffGen are comparable. We test the classification accuracy in two classifier algorithm respectively, and in each classifier there are four different private budget settings. In the test of ε budget, we define a series of DT height and for each height, we run 20 times and average the result over these runs.

1) Baseline: DiffGen and CDiffGen measure the Classification Accuracy(CA) on the anonymised testing set. In order to observe the impact of classification quality under the circumstances privacy budget joined, we adopt the Baseline Accuracy(BA) and Lower bound Accuracy(LA) as the bounds of proposed Classification Accuracy(CA). BA is the accuracy measured with raw data without $Specialization$ procedure to anonymization, so BA needn't satisfy the ε-differential privacy. The differences between BA and CA are the cost of privacy factor imported in the DiffGen/CDiffGen. LA is the accuracy measured with all of the attributes removed except for $class$. The difference between LA and CA is the benefit of DiffGen/CDiffGen over the nondisclosure approach. Therefore, the arithmetic value of BA - CA and CA - LA respectively represent the cost and benefit for DiffGen/CDiffGen to obey ε-differential privacy.

2) Task: . We get the BA and LA in advance. And then running DiffGen procedure to build a DT. With the same DT, DiffGen/CDiffGen produces the training/testing set for subsequent experiments. We evaluate the CA of DiffGen and CDiffGen with C4.5[14] classifier in two tasks, which differs in the utility function— the one is Information Gain and the other is Max[15].

B. Experiment Results

Figure 4 illustrate the Average Accuracy(CA) of DiffGen and CDiffGen in Information Gain utility function. The ε=0.1, 0.25, 0.5, 1, and the specialization[6](pre-defined DT height)=4, 7, 10, 13, 16. Figure 5 has the same experiment settings except that the utility function is Max. In the every subplot, the x-axis is the number of specialization, and the y-axis is classification accuracy rate. For each specialization, we run 20 times and average the result over these runs.

There are three trends from the the experiment results. First, the utility function Information Gain is not as good as Max. The score function is crucial for exponential mechanism to distinguish a 'good' or 'bad' attribute. If a utility function is suitable to dataset, the score can make a significant difference between different attributes.

Second, according to BA - CA and CA - LA, the results show a trade off between cost and benefit when adding the privacy budget. In DiffGen's curve, in contrast to the ε = 1, 0.5, 0.25, the interval of BA - CA is far smaller than that in the ε = 0.1, especially in utility function Max. For example in figure 5, when the specialization = 10, the BA - CA is 3.1%, 4.2%, 4.6% that corresponds to ε = 1, 0.5, 0.25, and the BA - CA is 7.5% in ε = 0.1, which is nearly twice than the others. These results indicate that the cost for achieving ε-differential privacy is small and growing slowly within a acceptable privacy budget limitation. This trend is suitable to CDiffGen when the specialization value is lower than 7, which benefit is larger than cost. But when the specialization value over 7, we could observe that CDiffGen is not compliant with this trend, which the benefit is significantly greater than cost, and including the subplot that ε = 0.1, the CA - LA is equally good.

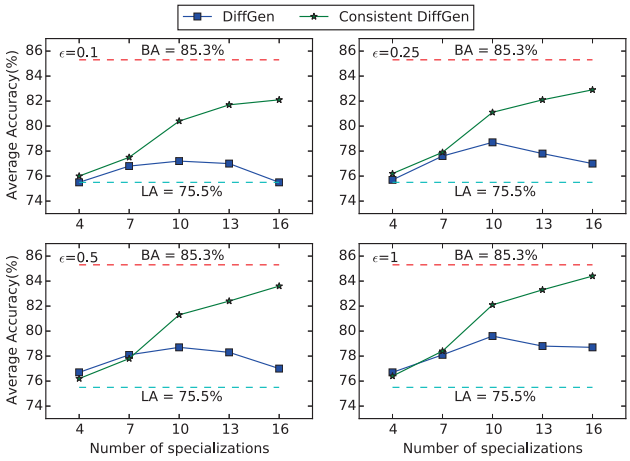


Fig. 4: Classification accuracy in Information Gain

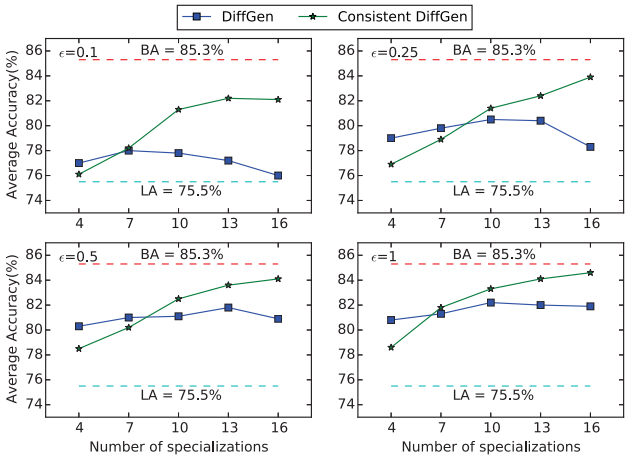


Fig. 5: Classification accuracy in Max

Third, in every subplot, the DiffGen's CA curve is up and down. These results present that specialization impacts CA in DiffGen algorithm. With the bigger specialization, the DT is more luxuriant, which leads to a better attribute partitioning, and hence, the CA increases. But when the specialization exceed a certain threshold, CA has obvious decrement with the increase of specialization, especially in Figure 4. For example, specailization = 10 is the threshold. With too many specializations, the nodes are subdivided into child nodes overly as well as the attached attribute counts. This situation leads to the extra dimension increasing on the final released data and makes the data sparse[16]. Thus, when the specialization exceed certain threshold, the summation of noise is quite likely to be closer to the raw counts, or even bigger than the raw counts, which makes the data useless.

On the CDiffGen's curve, we could observe that the curve is linear increasing trend in general. In addition to that, CA - LA is not affected by the ε value beyond a specialization threshold, in contrast to earlier discussion. This is the because that CDiffGen can sufficiently minimize the redundant noise while combining noisy counts. In most cases, CDiffGen can find a optimal set of collaborative nodes to cover the query scope. And noise-accumulation only happens among the elements in the set, which is directly proportional to set size. When specialization is small, the $Generalization$ procedure is weak to partition raw attributes and the boundaries of different entries in released data is vague, which makes CDiffGen and DiffGen indistinguishable.

V. Conclusion

As the discussion in experiments, the Consistent DiffGen effectively reduces the tension between noise variance and dimension of dataset in the matter of range-count queries. The CDiffGen curve almost grows linearly as the specializations increasing, which shows that the noise-accumulation makes little effect on it, just like the situation in lower specializations for DiffGen. It provides the most substantive evidence to what has been previously suspected that CDiffGen minimizes the redundant noise as far as possible during combining to answers. And it can adopt the simple data release mode.

Acknowledgment

This work was supported by Shanghai Key Laboratory in Shanghai JiaoTong University. We sincerely thank Alei Liang for multiple useful comments and suggestions, and thank the researchers for their excellent works.

References

[1] Sweney L. k-anonymity: A model for protecting privacy. International Journal on Uncertainty. Fuzines and Knowledge Based Systems.
[2] Machanavajhala A, Gehrke J, Kifer D, Venkitasubramaniam M. l-diversity: Privacy beyond k-anonymity. In ICDE.
[3] Wong R C W, Li J, Fu A W, Wang K.(α,k)-anonymity: An enhanced k-anonymity model for privacy-preserving data publishing. In SIGKD.
[4] Ganta S R, Kasiviswanathan S P, Smith A. Composition attacks and auxiliary information in data privacy. *Proceedings of the ACM SIGKDD*
[5] Wong R C W,Fu A,Wang K,et al. Can the utility of anonymized data be used for privacy breaches. *ACM Transactions on Knowledge Discovery from Data* ,2011,5(3):16
[6] N. Mohammed, R. Chen, B. C. M. Fung, and P. S. Yu. Differentially private data release for data mining. In *SIGKDD* , 2011.
[7] B. Barak, K. Chaudhuri, C. Dwork, S. Kale, F. McSherry, and K. Talwar. Privacy, accuracy, and consistency too: a holistic solution to contingency table release. PODS, 2007.
[8] C. Dwork, F. McSherry, K. Nissim, and A. Smith, Calibrating noise to sensitivity in private data analysis, in Proceedings of *TCC* , 2006.
[9] F. McSherry and K. Talwar. Mechanism design via differential privacy. In FOCS.
[10] Cormode G., Procopiuc C., Srivastava D., Tran T. T. Differentially private summaries for sparse data. In *ICDT*. ACM, 2012.
[11] X. Xiao, G. Wang, and J. Gehrke. Differential privacy via wavelet transforms. In *ICDE*, 2010.
[12] M.Hay, V.Rastogi, G.Miklau, D.Suciu, Boosting the accuracy of differentially-private queries through consistency. In Proceedings of *VLDB*, 2010.
[13] A. Frank and A. Asuncion. UCI machine learning repository, 2010.
[14] J. R. Quinlan. C4.5: Programs for Machine Learning. Morgan Kaufmann, 1993.
[15] L. Breiman, J. H. Friedman, R. A. Olshen, and C. J. Stone. Classification and Regression Trees. Wadsworth, 1984.
[16] G. Cormode, M. Procopiuc, D. Srivastava, and T. Tran, "Differentially private publication of sparse data", in Proceedings of *ICDT*, 2012.