Hindawi
Wireless Communications and Mobile Computing
Volume 2021, Article ID 8693978, 11 pages
https://doi.org/10.1155/2021/8693978

Research Article

Equation Chapter 1 Section 1 Differentially Private High-Dimensional Binary Data Publication via Adaptive Bayesian Network

Sun Lan, Jinxin Hong, Junya Chen, Jianping Cai, and Yilei Wang

College of Mathematics and Computer Science, Fuzhou University, Fuzhou 350108, China

Correspondence should be addressed to Yilei Wang; yilei@fzu.edu.cn

Received 17 April 2021; Accepted 21 June 2021; Published 17 July 2021

Academic Editor: Yong Zhang

When using differential privacy to publish high-dimensional data, the huge dimensionality leads to greater noise. Especially for high-dimensional binary data, it is easy to be covered by excessive noise. Most existing methods cannot address real high-dimensional data problems appropriately because they suffer from high time complexity. Therefore, in response to the problems above, we propose the differential privacy adaptive Bayesian network algorithm PrivABN to publish high-dimensional binary data. This algorithm uses a new greedy algorithm to accelerate the construction of Bayesian networks, which reduces the time complexity of the GreedyBayes algorithm from $O(nkC_{m+1}^{k+2})$ to $O(nm^4)$. In addition, it uses an adaptive algorithm to adjust the structure and uses a differential privacy Exponential mechanism to preserve the privacy, so as to generate a high-quality protected Bayesian network. Moreover, we use the Bayesian network to calculate the conditional distribution with noise and generate a synthetic dataset for publication. This synthetic dataset satisfies ε-differential privacy. Lastly, we carry out experiments against three real-life high-dimensional binary datasets to evaluate the functional performance.

1. Introduction

Various data are continuously collected and stored in different information systems with the continuous development of information technology. In the actual application process, people often encounter various data, such as medical, market trade, and travel track. These data usually have hundreds or thousands of attribute dimensions, and some are even higher. If these data are released, it may cause the leakage of sensitive personal information because high-dimensional data usually contain numerous personal privacy information. Therefore, this requires consideration of some measures to protect these information. However, protecting data privacy while ensuring data availability is a very challenging problem. The main reason is that the publishing space formed will grow exponentially as the attribute dimension increases.

The traditional privacy protection methods are mainly k-anonymity [1], (α, k)-anonymity [2], t-closeness [3], l-diversity [4], etc. However, all of these methods require special attack assumptions and knowledge background as support. It cannot be applied to general scenarios. Nevertheless, the data processed by techniques, such as differential privacy [5] and random disturbance [6, 7], do not need to make a series of conditional assumptions for the attacker and can be applied to various problem scenarios universally. Therefore, in recent years, such related technologies, especially differential privacy, have received increasing attention. Differential privacy is a typical data perturbation technique that perturbs information by adding noise that satisfies a specific distribution into the data. The disturbing data still retains the original statistical characteristics, but the attacker cannot reconstruct the real original data.

Many high-dimensional data publishing methods based on differential privacy are available, but these methods can only solve the problem to a certain extent, some problems still exist: first, these methods usually deal with the dimensional

disasters caused by high-dimensional data by converting them into low-dimensional data forcibly. This will cause serious information loss.

Second, the time complexity of these methods is generally too high. Although it can handle data of any dimension, in theory, it can only handle low dimension in the actual operation process. Because it takes such a long time, it cannot satisfy the need for higher-dimensional data.

Third, although most existing methods can handle high-dimensional binary data, it is easy for these methods to add too much noise, which leads to the data being completely covered, thus affecting the accuracy of publishing.

To address the challenges above, we propose the PrivABN algorithm, which is a high-dimensional binary data publishing method. Our main contributions are presented as follows:

(1) Instead of directly adding noise to the data, we use the Bayesian network method to avoid the impact of the dimensional disaster. In this way, the increase of global sensitivity with attribute dimension can be avoided, and the dimensional disaster can be solved effectively

(2) To reduce the total time complexity of the algorithm and enable it to process real high-dimensional data, a construction algorithm ABN is proposed by using a greedy algorithm, adaptive algorithm, and differential privacy index mechanism

(3) We propose a synthetic data generation algorithm SDG by using the characteristics of binary data and the topological order of the Bayesian network. This algorithm can reduce the magnitude of added noise and prevent excessive noise from covering the actual value

2. Materials and Methods

The materials and methods section should contain sufficient detail so that all procedures can be repeated. It may be divided into headed subsections if several methods are described.

3. Related Work

So far, many differential privacy publishing methods for high-dimensional data are available. Aiming at the privacy protection of high-dimensional binary data, Qardaji et al. [8] proposed the PriView method. This method assumes that all attributes are independent of each other and, then, answers user queries by constructing a set of low-dimensional noisy views, thereby reducing the impact of dimensional disasters. Zhang et al. [9, 10] proposed the PrivBayes method, which was directed at the issue of high-dimensional data privacy release. This method assumes that all attributes have a certain correlation and, then, constructed a Bayesian network between the attributes of the dataset by a greedy algorithm. Next, the Bayesian network is used to calculate the noisy joint distribution among attributes, and this joint distribution was utilized to generate a synthetic dataset for release. Based on the PrivBayes method, a series of derivative methods, such as weighted PrivBayes [11], Jtree [12], PrivHD [13], and PrivMN [14], has been proposed one after another. Wang et al. [11] set up a method for calculating attribute weights. They think that the importance of different attributes is different, and they will choose these important attributes first when building a Bayesian network. Chen et al. [12] used sparse vector sampling techniques to explore the relationships among attributes. Then, these relationships are used to build a Markov network (a special Bayesian network). Based on the Markov network, the joint tree algorithm is used to accelerate the solution of joint distribution, and the differential privacy protection is realized by adding Laplace noise to the joint distribution. Subsequently, Zhang et al. [13] introduced high-pass filtering technology based on the Jtree method to accelerate the construction of the Markov network. Finally, the PrivMN method, which is also based on the Markov network, is proposed by Wei et al. [14] to solve the joint distribution among attributes. The difference is that the approximate reasoning method is used in the calculation of the joint distribution.

The above analysis suggests that most of the existing methods consider how to construct the Bayesian or Markov network better to obtain a higher-precision joint distribution, and reducing publishing errors. However, these methods have high time complexity, which makes it impossible to process real high-dimensional data in practical applications. Moreover, the constructed Bayesian network still cannot reflect the true distribution well because of the degree's limitation. Therefore, this study proposes the PrivABN algorithm to solve the problems above.

4. Theorems and Definition

4.1. Differential Privacy

Definition 1 (ε-differential privacy). Let $D, D' \in \chi^d$ be two neighboring datasets, i.e., D and D' differ in only one record. Giving a randomized mechanism A, if A satisfies ε-differential privacy, the following is true:

$$\Pr\left[A(D) = O\right] \leq \exp\left(\varepsilon\right) \times \Pr\left[A\left(D'\right) = O\right], \quad (1)$$

where ε is the privacy budget and the smaller the privacy budget is, the higher the degree of privacy protection will be. $\Pr\left[A(D) = O\right]$ and $\Pr\left[A(D') = O\right]$ represent the probability that the algorithm A outputs as on the data set D and D', respectively.

Generally, Laplace [15] and Exponential mechanism [16] can realize differential privacy. Both of these mechanisms disturb the value or selection of the original data by generating noise. The magnitude of the generated noise is related to the global sensitivity of the query function.

Definition 2 (global sensitivity (see [15])). Let $f : D \longrightarrow \mathbb{R}^n$ be the query function. The global sensitivity is defined as

$$\Delta f = \max_{D,D'} \left\| f(D) - f\left(D'\right) \right\|_p, \tag{2}$$

where D and D' are two neighboring datasets and $\|.\|_p$ is the p-norm, which is a more commonly used 1-norm. Generally, the greater the global sensitivity is, the greater the noise generated by the mechanism and the impact on the algorithm results will be.

Theorem 1 (Laplace mechanism (see [15])). *Let $f : D \longrightarrow \mathbb{R}^n$ be the query function. Giving a randomized mechanism A, if A satisfies ε-differential privacy, the following is true:*

$$A(D) = f(D) + Lap(\Delta f/\varepsilon), \tag{3}$$

where $Lap(\Delta f/\varepsilon)$ is the noise variable that satisfies the Laplace distribution and where $Lap \sim Laplace(0, \Delta f/\varepsilon)$. Equation (3) shows that the larger the privacy budget ε or the smaller the global sensitivity Δf is, the smaller the noise generated will be.

Theorem 2 (Exponential mechanism (see [16])). *Let the score function $u(x)$ denote the score of x. Giving a random algorithm A, if A satisfies ε-differential privacy, the following is true:*

$$A(D) = \left\{ O_i \mid \Pr\left[A(D) = O_i\right] \in \exp\left(\frac{\varepsilon u(O_i)}{2\Delta u}\right) \right\}, \tag{4}$$

where Δu is the global sensitivity of the score function $u(x)$. This formula means that for each output result O_i of algorithm A, a probability of $\exp\left(\varepsilon \cdot u(O_i)/(2\Delta u)\right)$ *being selected is likely to exist. The higher the score of O_i is, the greater the probability of being selected will be.*

In addition, in designing and proving to meet the differential privacy algorithm, an important combination of differential privacy needs to be used.

Property 1 (sequential composition (see [17])). Giving a dataset D and a set of differential privacy algorithms $A_1(D), A_2(D), \cdots, A_m(D)$ and the algorithm $A_i(D)$ satisfies ε_i-differential privacy. Moreover, the random processes of any two algorithms are independent of each other. Then, the combination of these algorithms A satisfies $\sum_{i=1}^{m} \varepsilon_i$-differential privacy.

4.2. Bayesian Network. Bayesian network is a probabilistic graph model, mainly used to explore the relationship between a group of objects. Usually, a directed acyclic graph is used to represent the Bayesian network. The nodes in the graph represent objects, and the edges represent relationships.

In general, giving a set of attributes set $S = \{S_1, S_2, \cdots, S_m\}$, its joint distribution can be expressed as

$$\Pr[S] = \Pr[S_1] \cdot \cdots \cdot \Pr[S_m \mid S_1, \cdots, S_{m-1}]. \tag{5}$$

Through the Bayesian network constructed by the attribute set, its joint distribution can be approximated as

$$\Pr[S] \approx \Pr_{\mathcal{N}}[S] = \prod_{i=1}^{m} \Pr[S_i \mid \Pi_i], \tag{6}$$

where Π_i is the parent node set of node S_i. If the constructed Bayesian network can represent the relationship between attributes well, then $\Pr_{\mathcal{N}}[S] \longrightarrow \Pr[S]$.

Therefore, how to build a better Bayesian network is important.

4.3. Conditional Entropy. Conditional entropy can be used to measure the degree of interdependence among attributes. The larger the value, the higher the degree of dependence between attributes.

Definition 3 (conditional entropy). Giving two discrete random variables $X \in \{x_1, x_2, \cdots, x_n\}$ and $Y \in \{y_1, y_2, \cdots, y_m\}$, the conditional entropy between them is

$$I(X, Y) = \sum_{i=1}^{n} \sum_{j=1}^{m} \Pr\left[x_i, y_j\right] \mathrm{lb} \frac{\Pr\left[x_i, y_j\right]}{\Pr[x_i] \Pr\left[y_j\right]}, \tag{7}$$

where $\Pr[x_i, y_j]$ is the joint distribution probability value of $X = x_i$ and $Y = y_j$.

Equation (7) shows that when $I(X, Y) \longrightarrow 0$, there is $\Pr[x_i, y_j] \longrightarrow \Pr[x_i] \Pr[y_j]$, that is, variables X and Y are close to independent of each other.

5. The PrivABN Algorithm

In Table 1, the meanings of the commonly used symbols in this section are provided, and the other symbols are explained when used.

5.1. Differential Privacy Bayesian Network Algorithm. Zhang et al. [9] proposed a conditional entropy-based degree Bayesian network construction algorithm GreedyBayes in PrivBayes. The main idea of this algorithm is to select a pair of the largest conditional entropy to join the current Bayesian network each time.

The GreedyBayes algorithm is a common algorithm used to construct Bayesian networks, and its implementation is shown in Algorithm 1.

Considering that Zhang et al. [9, 10] did not provide the time complexity formula of the algorithm in the article, this study demonstrates the time complexity of the GreedyBayes algorithm.

Table 1: Table of notations.

Notation	Description
D	Original data set
$\tilde{D}$	Synthetic data set
S	D's attribute set
n	Number of records in D
m	Number of attributes of D
$\mathcal{N}$	Bayesian network

Theorem 3. *The time complexity of the GreedyBayes algorithm is* $O(nkC_{m+1}^{k+2})$.

Proof. The time consumption of the GreedyBayes algorithm is mainly concentrated in the for loop of Step 3. The for loop is executed a total of $m-1$ times, and each time $|S \setminus V| \cdot C_V^k$ pairs of (S_i, Π_i) are generated. Therefore, a total of

$$\begin{aligned}&(m-1)\cdot C_1^1+\cdots+(m-d)\cdot C_k^k+\cdots+C_{m-1}^k\\&=\frac{(2m-k-2)(k-1)}{2}+C_{k+1}^{k+1}+C_{k+2}^{k+1}+\cdots+C_m^{k+1},\\&=\frac{(2m-k-2)(k-1)}{2}+C_{m+1}^{k+2}.\end{aligned}\tag{8}$$

(S_i, Π_i) pairs will be generated in the whole process. In the worst case, when$k+2=(m+1)/2$, any $k \in \mathrm{N}_+$ holds for

$$\begin{aligned}\frac{(2m-k-2)(k-1)}{2}&=\frac{(3k+4)(k-1)}{2}<C_{m+1}^{k+2}\\&=\frac{(2\cdot(k+2))!}{((k+2)!)^2}\\&=\frac{(k+3)\cdot(k+4)\cdots(2k+3)\cdot(2k+4)}{1\cdot 2\cdots(k+1)\cdot(k+2)}.\end{aligned}\tag{9}$$

For each (S_i, Π_i) pair, the conditional entropy size needs to be calculated, and each calculation takes $O(nk)$ time. Therefore, the total time complexity of the algorithm is $O(nkC_{m+1}^{k+2})$.

Evidently, due to the influence of time complexity, the GreedyBayes algorithm can only be applied when the number of attributes m is small or the maximum degree k of the Bayesian network is small.

To process real high-dimensional data, a more complex Bayesian network is constructed. This paper proposes a simple and efficient Bayesian network construction algorithm ABN. This algorithm only needs the time complexity of $O(nm^4)$ to construct a complete Bayesian network.

In order to more intuitively illustrate the advantages of the ABN algorithm in time performance, we take a dataset containing 50 attributes as an example. When the maximum degree k of the Bayesian network is 5, 10, 15, 20, 25, the number of (S_i, Π_i) pairs enumerated by the GreedyBayes algorithm and ABN algorithm are shown in Table 2.

It can be seen that, assuming, the computer can calculate the mutual information of 10,000 (S_i, Π_i) pairs per second. When the maximum degree $k=10$, it already takes 184 days of uninterrupted processing to complete. When $k=25$, even the computer cannot solve it, because it requires a total of 728 years and 11 days of uninterrupted processing to complete the task. However, the ABN algorithm can find a relatively good (S_i, Π_i) pair no matter how much k value is; it only takes 250 times. This is very valuable in practical application.

The specific implementation of the ABN algorithm is shown in Algorithm 2.

In addition to solving the problem of the GreedyBayes's execution efficiency, the ABN algorithm also introduces an Exponential mechanism of differential privacy to disturb the Bayesian network construction process to solve privacy leakage caused by the Bayesian network.

It can be seen from the algorithm flow that the ABN algorithm removes the limitation of the maximum degree of the Bayesian network. And adopt the way of adaptively selecting the number of degree of each node in the network, which makes the network structure become more complex and diverse. The resulting network contains more information, and the synthetic data generated from the network is more likely to match the real data.

The main difference between the ABN algorithm and the GreedyBayes algorithm is that the former uses a greedy algorithm GParentSet with time complexity of $O(nm^2)$ to solve the optimal parent attribute set under each in-degree of the current attribute S_i. Compared with the ABN algorithm, the GreedyBayes algorithm traverses the values of all parent attribute sets through brute force enumeration every time it searches for the optimal parent attribute set. Actually, in this process, a lot of repeated and useless calculations are performed. Therefore, the ABN algorithm adopts the memorization, and its characteristics are suitable for the optimal substructure. On the premise of ensuring the best solution, it can greatly reduce the repetitive and useless calculation process and improve the construction speed of the Bayesian networks.

The specific implementation of the GParentSet algorithm is shown in Algorithm 3.

In this algorithm, $dp[S, i]$ is a set of attributes, which indicates that the current attribute S has selected i attributes as the best choice of parent attributes. In fact, when the GParentSet algorithm is executed in the new round, $dp[S, i]$ has recorded the optimal parent set selection in this state in the previous round. Therefore, for this round, we only need to care about the selection of the newly added parent attribute fa in the previous round.

In the ABN algorithm, in Step 5, each candidate attribute S_i and its optimal parent attribute set $\{\Pi_{i,1},\cdots,\Pi_{i,i\text{-}1}\}$ under various in-degree values are all added to the set Ω. In Step 6, it is selected through the differential privacy Exponential mechanism. This process does not limit the maximum in-degree of the Bayesian network. Among all the optional degrees, the degree with the greater amount of

Input: data set D, attribute set S, maximum in-degree k
Output: synthetic data set $\tilde{D}$
1: Initialize $\mathcal{N} = \varnothing$, $V = \varnothing$;
2: Randomly select an attribute from the attribute set S as S_1, add $(S_1, \varnothing)$ to$\mathcal{N}$, and add S_1 to the vertex set V;
3: for $i = 2$ to m do
4: Initialize collection $\Omega = \varnothing$;
5: For each $S_i \in S/V$ and $\Pi_i \in \binom{V}{k}$, add (S_i, Π_i) to Ω;
6: Select (S_i, Π_i) with the largest mutual information from Ω and join $\mathcal{N}$, add S_i to V;
7: end for
8: return $\mathcal{N}$.

Algorithm 1: GreedyBayes Algorithm.

Table 2: Eumeration times of Greedybayes algorithm and ABN algorithm.

k	C_{m+1}^{k+2}	m^2
5	$\approx 1.16 \times 10^8$	=250
10	$\approx 1.59 \times 10^{11}$	=250
15	$\approx 1.48 \times 10^{13}$	=250
20	$\approx 1.56 \times 10^{14}$	=250
25	$\approx 2.30 \times 10^{14}$	=250

conditional entropy is easier to be selected. We do not need to control the structure of the resulting Bayesian network. Its construction is an adaptive process toward greater conditional entropy. This method is more flexible and reasonable than the traditional method of constructing Bayesian networks that requires a fixed network's maximum in-degree. Considering that the maximum in-degree of the Bayesian network constructed by this method is not fixed, it will adjust adaptively according to different datasets. Moreover, it will adjust in the direction of making the conditional entropy of the entire Bayesian network larger.

The ABN algorithm uses the differential privacy Exponential mechanism in Step 6. Its idea is to select the (S, Π) pairs currently added to the Bayesian network based on probability. We use conditional entropy I as the scoring function of the Exponential mechanism $u(S, \Pi) = I(S, \Pi)$. The greater the conditional entropy of the (S, Π) air, the higher the scoring function value and the greater the probability of being selected.

Zhang et al. [10] proved in the article that the global sensitivity of conditional entropy on binary data is

$$\Delta I = \frac{1}{n}\mathrm{lb}n + \frac{n-1}{n}\mathrm{lb}\frac{n}{n-1}. \tag{10}$$

Although they further gave a scoring function F with lower global sensitivity in the article, its global sensitivity is $\Delta F = 1/n$. However, the time complexity required to calculate the scoring function is $O(n2^m)$, which can only be applied when the attribute dimension m is small. Therefore, we do not adopt this method and still uses conditional entropy as the scoring function.

Step 6 will be executed $m - 1$ times, each time the Exponential mechanism is used to select a (S, Π) pair from Ω to join the Bayesian network. From Property 1, we know that each choice needs to consume a part of the privacy budget. Here, we use the average division method to allocate the privacy budget, that is, ε_1 is divided equally into $m - 1$ shares. Then, combined with the Exponential mechanism, we can obtain the expression of $\mathrm{Pc}(S, \Pi)$ as

$$\mathrm{Pc}(S, \Pi) = \frac{\exp\left(\varepsilon_1 u(S, \Pi)/(2(m-1)\cdot\Delta u)\right)}{\sum_{(S_i,\Pi_i)\in\Omega}\exp(\varepsilon_1 u(S_i, \Pi_i)/(2(m-1)\cdot\Delta u))}. \tag{11}$$

Finally, proving that the ABN algorithm satisfies ε_1-differential privacy. The final output of the algorithm is a Bayesian network $\mathcal{N}$. In Step 6, the Exponential mechanism is used to select the currently added attribute node and its parent attribute set node. This operation disturbs the construction of the Bayesian network. According to Theorem 2, this step satisfies ε_1-differential privacy. Moreover, the entire ABN algorithm satisfies ε_1-differential privacy because no other operations involve the use of the original dataset D.

5.2. Differential Privacy Synthetic Data Release. The Bayesian network can simplify the calculation of the joint distribution between attributes to a large extent, and the better the Bayesian network is, the closer the joint distribution is to the true value. However, if the Bayesian network is directly used to calculate the joint distribution between attributes, it may still cause privacy leakage. Therefore, we need to perturb the calculated joint distribution further to achieve the purpose of protecting privacy.

Zhang et al. [9] used the NoisyConditionals algorithm to realize the secure calculation of Bayesian networks. This algorithm adds Laplace noise into the joint distribution $\Pr[S, \Pi]$ to obtain the joint distribution $\Pr^*[S, \Pi]$ with noise. Although this algorithm can ensure that the obtained joint distribution meets the differential privacy protection, its joint distribution may become very sparse, and the original probability value will generally be small when the

Input: data set D, attribute set S, privacy budget ε_1
Output: Bayesian network $\mathcal{N}$
1: Initialize $\mathcal{N} = \varnothing$, $V = \varnothing$;
2: Randomly select an attribute from the attribute set S as S_1, add $(S_1, \varnothing)$ to $\mathcal{N}$, and S_1 to V, $fa = S_1$;
3: for $i = 2$ to m do
4: Initialize collection $\Omega = \varnothing$;
5: For each, find $\Pi_i = \{\Pi_{i,1}, \cdots, \Pi_{i,i\text{-}1}\} \longleftarrow$ GParentSet(S_i, fa, i) and then add $(S_i, \Pi_{i,1}), \cdots, (S_i, \Pi_{i,i-1})$ to Ω;
6: Use the Exponential mechanism with privacy budget $\varepsilon_1/(m-1)$ to select a $(S_i, \Pi_{i,j})$ from Ω to add to $\mathcal{N}$ with probability Pc(S, Π), and add S_i to V, $fa = S_i$;
7: end for
8: return $\mathcal{N}$.

Algorithm 2: ABN Algorithm.

Input: current attribute S, new parent attribute fa, maximum in-degree k
Output: optimal parent attribute set $\Pi = \{\Pi_1, \cdots, \Pi_k\}$
1: Initialize $\Pi = \varnothing$;
2: for $i = 1$ to k do
3: $\Pi' = dp[S, i-1] \cup fa$;
4: $\Pi'' = dp[S, i]$;
5: $dp[S, i] = \begin{cases} \Pi' & I(S, \Pi') \geq I(S, \Pi'') \\ \Pi'' & I(S, \Pi') < I(S, \Pi'') \end{cases}$;
6: Add $dp[S, i]$ to Π;
7: end for
8: return Π.

Algorithm 3: GParentSet Algorithm.

attribute dimension increases. At this time, if Laplace noise is directly added into these smaller probability values, then it may cause the problem that the noise completely covers the true value, and seriously affecting the accuracy of the release.

Therefore, we do not directly use a joint distribution like the NoisyConditionals algorithm. Instead, we use conditional distribution to generate synthetic data because of the characteristics of binary data with only 0 and 1. The main reason is that when the distribution is a conditional distribution, we have the following equation:

$$\Pr[S_i = 0 \mid \Pi_i] + \Pr[S_i = 1 \mid \Pi_i] = 1. \quad (12)$$

According to the equation, we can infer that at least one relatively large conditional probability value exists in the conditional distributions $\Pr[S_i = 0 \mid \Pi_i]$ and $\Pr[S_i = 1 \mid \Pi_i]$. In this way, if noise is directly added into the conditional probability, then at least one larger conditional probability will not be covered by the noise. Even if the conditional probability is completely covered by noise, it has minimal effect on the accuracy of the final release. The reason is that if such a conditional probability exists, then its probability value must be very small or negligible relative to another probability value. Therefore, even after normalization, they can still reflect the original distribution law.

Nevertheless, synthetic data can still be generated. The traditional method is to generate it directly through joint distribution, but it still has the same problems as data sparseness. So this study uses conditional distribution to generate synthetic data. To this end, we design a synthetic data generation algorithm SDG. The main idea of the SDG algorithm is to use the conditional distribution of the node and its parent node to generate synthetic data according to the topological order of the Bayesian network.

The specific implementation of the SDG algorithm is shown in Algorithm 4.

The synthetic data $\tilde{D}$ generated by the SDG algorithm can make the attacker unable to infer a specific record in the original dataset D; thus, it protects the personal privacy information in the data from being leaked.

Finally, proving that the SDG algorithm satisfies ε_2-differential privacy. The algorithm adds Laplace noise Lap$(2m/(n \cdot \varepsilon_2))$ to each conditional distribution $\Pr[S_i \mid \Pi_i]$ in Step 5 and obtains the conditional distribution $\Pr^*[S_i \mid \Pi_i]$ with noise. Then, these conditional distributions are used to generate synthetic data for release. According to Theorem 2, this process satisfies ε_2-differential privacy. Moreover, because no other subsequent steps involve the use of the original data set D, the entire SDG algorithm satisfies ε_2-differential privacy.

5.3. Differential Privacy Adaptive Bayesian Network Algorithm. The PrivABN algorithm can be divided into two independent steps:

(1) Through the ABN algorithm, construct a Bayesian network $\mathcal{N}$ which satisfies differential privacy for

Input: data set D, Bayesian network $\mathcal{N}$, privacy budget ε_2
Output: synthetic data set $\tilde{D}$
1: Initialize $P^* = \varnothing$;
2: for $i = 1$ to m do
3: Calculate the joint distribution $\Pr[S_i, \Pi_i]$ and the marginal distribution $\Pr[\Pi_i]$;
4: Calculate conditional distribution $\Pr[S_i \mid \Pi_i] = \Pr[S_i, \Pi_i]/\Pr[\Pi_i]$;
5: Add Laplace noise $\mathrm{Lap}(2m/n \cdot \varepsilon_2)$ to $\Pr[S_i \mid \Pi_i]$ to get $\Pr^*[S_i \mid \Pi_i]$;
6: Reset the negative value in $\Pr^*[S_i \mid \Pi_i]$ to 0, normalize other values, and then addP^*;
7: end for
8: Traverse the node S_i according to the topological order of the Bayesian network $\mathcal{N}$;
9: Obtain the noisy conditional distribution $\Pr^*[S_i \mid \Pi_i]$ from P^*, and update the value of each record attribute S_i in $\tilde{D}$ according to the conditional distribution;
10: return $\tilde{D}$.

Algorithm 4: SDG Algorithm.

the original dataset D, and extract the noise conditional distribution P^* from the Bayesian network

(2) Through the SDG algorithm, generate a synthetic dataset $\tilde{D}$ for release, according to the topological order of the Bayesian network $\mathcal{N}$ and the conditional distribution P^* with noise

The specific implementation process of the PrivABN algorithm is shown in Algorithm 5.

In the PrivABN algorithm, the subalgorithms ABN and SDG involve the use of the original dataset D. According to property 1, in order for the PrivABN algorithm to satisfy ε-differential privacy, privacy budgets must be allocated for these two subalgorithms first. According to the analysis in the previous chapters of this paper, the privacy budget ε_1 and ε_2 will, respectively, correspond to the two noise distributions $\exp(\varepsilon_1 u(S, \Pi)/(2(m-1) \cdot \Delta u))$ and $\mathrm{Lap}(2m/(n \cdot \varepsilon_2))$. The first noise distribution aims to make the value of $\varepsilon_1/(2(m-1) \cdot \Delta u)$ (with the nondependent $u(S, \Pi)$) as large as possible. The second noise distribution aims to make the value of $2m/(n \cdot \varepsilon_2)$ as small as possible. By simplifying the two formulas above, we can obtain

$$\begin{aligned} &\max\left(\frac{\varepsilon_1}{2(m-1)\cdot\Delta u}\right) \\ &= \max\left(\varepsilon_1 \cdot \frac{1}{2(m-1)\cdot((1/n)\mathrm{lb}n + (n-1/n)\mathrm{lb}(n/n-1))}\right) \\ &\approx \max\left(\varepsilon_1 \cdot \frac{n}{2(m-1)}\right), \end{aligned}$$

$$\min\left(\frac{2m}{n\cdot\varepsilon_2}\right) = \min\left(\frac{1}{\varepsilon_2}\cdot\frac{2m}{n}\right) = \max\left(\varepsilon_2 \cdot \frac{n}{2m}\right). \quad (13)$$

From the simplified results, the proportions of the two privacy budgets that need to be allocated are roughly the same. Therefore, we adopt an even distribution strategy to allocate the privacy budget ε.

Finally, proving that the PrivABN algorithm satisfies ε-differential privacy. It can be seen that the ABN algorithm and SDG algorithm satisfy ε_1-differential privacy and ε_2-differential privacy. Apart from the above two algorithms, the PrivABN algorithm has no other place that involves the use of the original data set D. Therefore, according to property 1, the PrivABN algorithm satisfies $(\varepsilon_1 + \varepsilon_2)$-differential privacy.

6. Experiences

6.1. Experiences Environment. The experimental platform is a 4-core Intel i5-6300HQ CPU (2.3GHz), 8GB memory, Windows 10 operating system, and the compilation environment is Dev-C++5.11. Our experiments use the C++ programming language to implement all the methods, among which the implementation of the Bayesian network refers to the relevant code of the paper experiment by Zhang et al. [10].

6.2. Datasets. Our experiments use three real-world datasets, NLTCS, ACS, and Retail. NLTCS [18] is an American long-term care survey record that includes the daily life and medical conditions of 21,574 elderly disabled persons. ACS [19] is the global census data released by IPUMS-USA, which records 47,461 pieces of personal information. Retail [20] is 88,162 shopping records in the US retail market. Each record contains items purchased by it, a total of 16,469 categories of goods, from which we have retained the top 50 best-selling goods. The specific information of these three data sets is shown in Table 3.

6.3. Evaluation. For each set of experiments, we will compare the $L1$ error (mean error) between the generated synthetic dataset $\tilde{D}$ and the original dataset D. Moreover, the $L2$ error is caused by the same number of α-way queries on these two datasets.

Definition 4 ($L1$ error). The $L1$ error between the original dataset D and the synthetic dataset $\tilde{D}$ is

$$L_1(D, \tilde{D}) = \frac{\sum_{i=1}^{N}\sum_{j=1}^{M}\left|D_i^{(j)} - \tilde{D}_i^{(j)}\right|}{N}, \quad (14)$$

Input: dataset D, attribute set S, privacy budget ε
Output: synthetic data set $\tilde{D}$
1: $\varepsilon_1 = \varepsilon/2, \varepsilon_2 = \varepsilon/2$;
2: $\mathcal{N} \longleftarrow \mathrm{ABN}(D, S, \varepsilon_1)$;
3: $\tilde{D} \longleftarrow \mathrm{SDG}(D, \mathcal{N}, \varepsilon_2)$;
4: return $\tilde{D}$.

Algorithm 5: PrivABN Algorithm.

Table 3: Description of datasets.

Dataset	Type	Cardinality	Dimensionality	Domain size
NLTCS	Binary	21 574	16	2^{16}
ACS	Binary	47 461	23	2^{23}
Retail	Binary	88 162	50	2^{50}

where $D_i^{(j)}$ is the value of the jth row of the ith experiment using the original dataset and $\tilde{D}_i^{(j)}$ is the value of the jth row of the ith experiment using the synthetic dataset. N is the number of experiments, and M is the number of rows of the dataset.

Definition 5 ($L2$ error). Let the α-way edge table generate by the original dataset D through the ith query is $T_i(D)$, and the α-way edge table generate by the synthetic dataset $\tilde{D}$ is $T_i(\tilde{D})$; the $L2$ error between them is

$$L_2\left(D, \tilde{D}\right) = \frac{\sum_{i=1}^{N} \sqrt{\sum_{j=1}^{M} \left(T_i^{(j)}(D) - T_i^{(j)}\left(\tilde{D}\right)\right)^2}}{N}, \quad (15)$$

where $T_i^{(j)}$ is the value of the jth row of the α-way edge table generated by the ith query, N is the number of queries, and M is the number of rows of the edge table.

6.4. Result Analysis. The first part of the experiments is to analyze the availability of the PrivABN method. To know the size of the noise generated by the PrivABN method in a noise-free environment, we set up the NoPrivABN method without differential privacy protection for comparison. We will conduct 100 random repeated experiments on NLTCS and ACS. We set the privacy budget ε to 0.001, 0.005, 0.01, 0.05, 0.1, and 1.0. The experimental results will be verified by 200 random α-way queries, the values of which are 3, 7, and 12. We will test their performance on $L1$ error and $L2$ error. The experimental results are shown in Figure 1.

Figure 1 shows that the PrivABN algorithm only needs a very small privacy budget. When $\varepsilon = 0.05$, it is very close to the effect of NoPrivABN, which shows that the PrivABN algorithm has high availability.

Moreover, as the privacy budget ε increases, the error generated by PrivABN gradually approaches the error of NoPrivABN, which is in line with the differential privacy law. This finding further verified the credibility of the experiment.

Another finding is that the PrivABN algorithm without differential privacy protection will produce certain errors. The reason is that the algorithm itself generates a synthetic dataset based on the Bayesian network and conditional distribution. The process of constructing the Bayesian network itself may produce certain errors, and the process of generating synthetic data through conditional distribution is probabilistic, thereby resulting in the production of certain errors. Therefore, taking the error of NoPrivABN as the lower bound is in line with the experimental standard.

Finally, the performance of PrivABN on different α-way queries under the same dataset is observed. Their overall trend of change and their turning points are also the same. Their error sizes under different privacy budgets do not differ greatly, and they are basically the same within a certain error range. Therefore, we can consider that the stability of PrivABN in the face of different conditional parameters is a manifestation of the PrivABN's remarkable robustness.

According to the $L1$ errors and $L2$ errors between the generated synthetic dataset and the original dataset, PrivABN only needs a very small privacy budget to achieve the effect of NoPrivABN. To know this threshold more accurately, we further subdivide the value of the privacy budget. We set the privacy budget to increase from 0.05 to 0.5 in increments of 0.025 and from 0.5 to 1.0 in increments of 0.1. Then, experiments with 3-way query under the NLTCS and ACS datasets, respectively (from the previous experimental conclusions, it can be known that PrivABN has better robustness, and its error under different α-way queries is not much different, so only one query needs to be compared). The experimental results are shown in Figure 2.

Figure 2 shows that when the privacy budget is 0.4 (NLTCS dataset) and 0.225 (ACS dataset), the $L2$ error is lower than the error bar of 0.01. This finding shows that the PrivABN method only needs to consume a very small privacy budget to achieve good privacy protection. Therefore, we can definitely believe that privabn has high availability.

From another point of view, the PrivABN algorithm only needs to give a small amount of privacy budget. It can achieve a good privacy protection effect and can greatly reduce the error of differential privacy protection. This is the reason for its high availability.

In the second part of the experiments, the performance of PrivABN on the real high-dimensional dataset Retail is analyzed. To reflect the pros and cons of the results better, the experiments will be compared with these three methods, namely, PriView [8], PrivBayes [9], and Jtree [12]. We will conduct 100 repeated random experiments on the Retail dataset and set the privacy budget ε to 0.1 and 1.0. The experimental results will be verified by 200 random α-way queries, where α values correspond to 4, 6, and 8. We will test their performance on $L2$ errors. The experimental results are shown in Figure 3.

Figure 3 shows that under different privacy budgets, the PrivABN method performs significantly better than the three other methods on the Retail dataset. Further, when the privacy budget is small ($\varepsilon = 0.1$), the PrivABN method performs

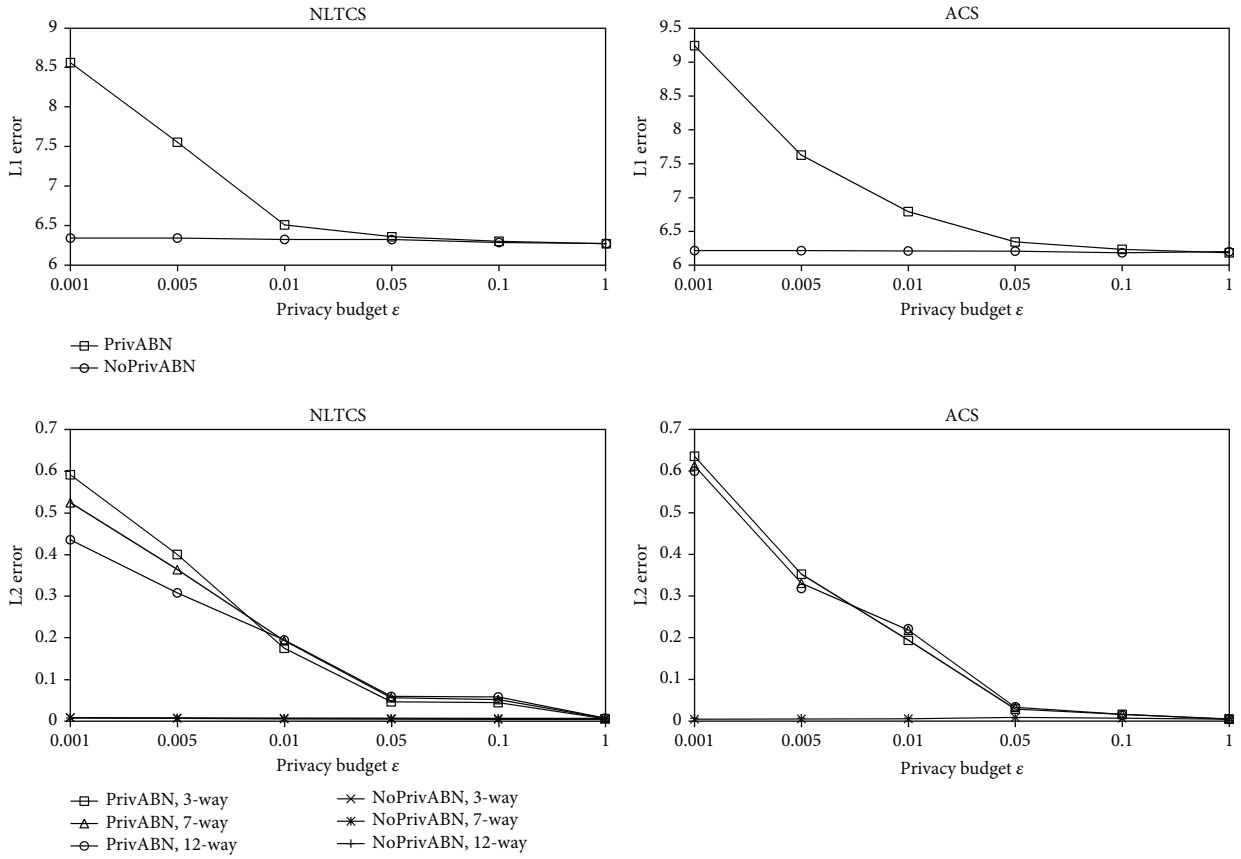

Figure 1: Error analysis with or without privacy protection—1.

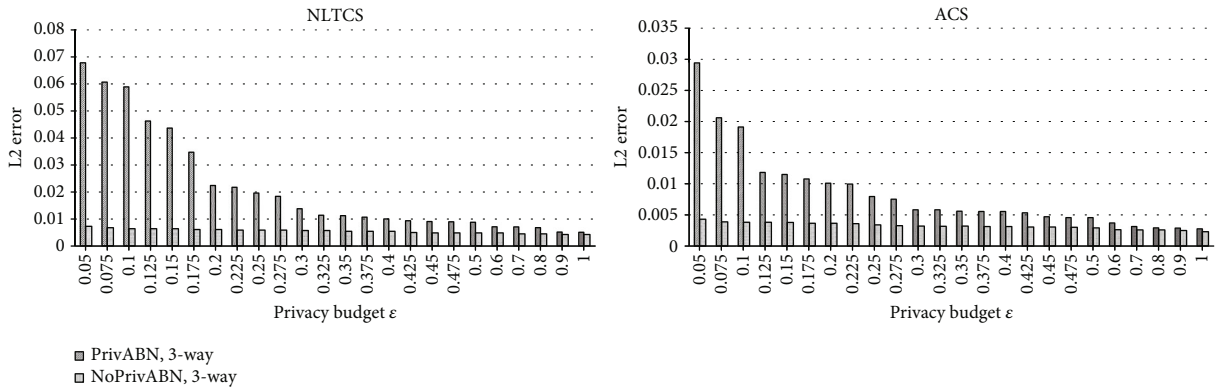

Figure 2: Error analysis with or without privacy protection—2.

significantly better than the other three methods. When the privacy budget is large ($\varepsilon = 1.0$), PrivABN is still superior to the other three methods, but it is not different from the JTree method. This is in line with the law of differential privacy, because as the privacy budget continues to increase, the degree of privacy protection of the algorithm will continue to decline, until the implementation result is consistent with that of the algorithm without differential privacy protection. Therefore, it can be seen that the accuracy of the probabilistic graph model itself built by the Privabn method is better than that of the model built by the JTree method. Compared with the model built by the PrivBayes method, the accuracy is much better. This further verifies that the improvement strategies proposed in this paper are effective and have achieved good results.

Moreover, with the increase in α-way query dimension, the variation range of $L2$ error of the PrivABN method is significantly smaller than those of the three other methods. Therefore, we can further infer that the PrivABN method has higher availability and better robustness.

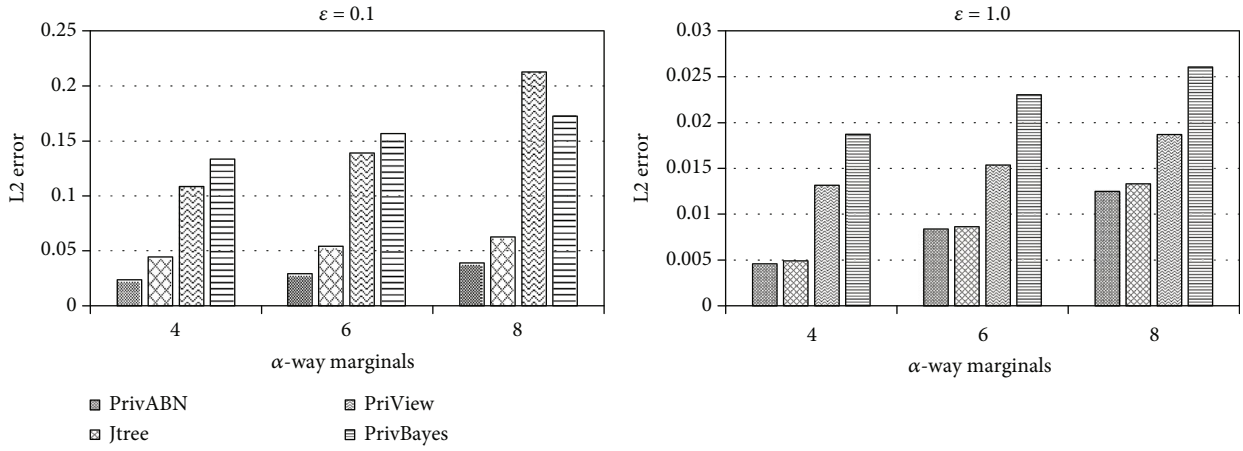

Figure 3: Error analysis of α-way query in different methods on Retail dataset.

7. Conclusion

Private releasing of high-dimensional data has been a research hotspot and a challenge in the field of differential privacy. This study proposes an efficient and low-noise differential privacy publishing method called PrivABN for high-dimensional binary data. The method uses the ABN algorithm to construct the Bayesian network over the dataset quickly and adaptively while using the differential privacy Exponential mechanism to protect the privacy of the Bayesian network during the construction. Subsequently, the SDG algorithm uses the differential privacy Laplace mechanism to initially extract the noisy conditional distribution from the Bayesian network and then uses these conditional distributions and the Bayesian network topology to generate synthetic data for release. By performing experiments on three real data sets, we demonstrate that PrivABN deserves higher usability and robustness than existing methods.

The main focus for our future work will be continually on the differential privacy publication of high-dimensional data. We will investigate the differential privacy publication of high-dimensional nonbinary data and explore the issue of differential privacy publication in a streaming high-dimensional data environment.

Data Availability

NLTCS is an American long-term care survey record that includes the daily life and medical conditions of 21,574 elderly disabled persons. ACS is the global census data released by IPUMS-USA, which records 47,461 pieces of personal information. Retail is 88,162 shopping records in the US retail market.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

References

[1] L. Sweeney, "k-anonymity: a model for protecting privacy," *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, vol. 10, no. 5, pp. 557–570, 2002.

[2] R. C.-W. Wong, J. Li, A. W.-C. Fu, and K. Wang, "(α, k)-anonymity: an enhanced k-anonymity model for privacy preserving data publishing," in *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 754–759, Philadelphia, USA, 2006.

[3] N. Li, T. Li, and S. Venkatasubramanian, "t-closeness: privacy beyond k-anonymity and l-diversity," in *2007 IEEE 23rd International Conference on Data Engineering*, pp. 106–115, Istanbul, Turkey, 2007.

[4] A. Machanavajjhala, D. Kifer, J. Gehrke, and M. Venkitasubramaniam, "*L*-diversity: privacy beyond *k*-anonymity," *ACM Transactions on Knowledge Discovery from Data (TKDD)*, vol. 1, no. 1, p. 3, 2007.

[5] C. Dwork, "Differential privacy," 2006.

[6] K. Muralidhar and R. Sarathy, "Security of random data perturbation methods," *ACM Transactions on Database Systems*, vol. 24, no. 4, pp. 487–493, 1999.

[7] H. Kargupta, S. Datta, Q. Wang, and K. Sivakumar, "On the privacy preserving properties of random data perturbation techniques," in *Data Mining, 2003. ICDM 2003. Third IEEE International Conference on*, Florida, USA, 2003.

[8] W. Qardaji, W. Yang, and N. Li, "Priview: practical differentially private release of marginal contingency tables," in *Proceedings of the 2014 ACM SIGMOD international conference on Management of data*, pp. 1435–1446, UT, USA, 2014.

[9] J. Zhang, G. Cormode, C. M. Procopiuc, D. Srivastava, and X. Xiao, "*PrivBayes: private data release via bayesian networks*," 2014.

[10] J. Zhang, G. Cormode, C. M. Procopiuc, D. Srivastava, and X. Xiao, "Privbayes: private data release via bayesian networks," *ACM Transactions on Database Systems (TODS)*, vol. 42, no. 4, pp. 1–41, 2017.

[11] W. Liang, W. Weiping, and M. Dan, "Privacy preserving data publishing via weighted bayesian networks," *Journal of Computer Research and Development*, vol. 53, no. 10, article 2343, 2016.

[12] R. Chen, Q. Xiao, Y. Zhang, and J. Xu, "Differentially private high-dimensional data publication via sampling-based inference," in *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 129–138, Sydney, NSW, Australia, 2015.

[13] Z. Xiaojian, C. Li, J. Kaizhong, and M. Xiaofeng, "Private high-dimensional data publication with junction tree," *Journal of Computer Research and Development*, vol. 55, no. 12, article 2794, 2018.

[14] F. Wei, W. Zhang, Y. Chen, and J. Zhao, "Differentially private high-dimensional data publication via markov network," in *International Conference on Security and Privacy in Communication Systems*, pp. 133–148, Singapore, 2018.

[15] C. Dwork, F. McSherry, K. Nissim, and A. Smith, "Calibrating noise to sensitivity in private data analysis," in *Theory of cryptography conference*, pp. 265–284, New York, NY, USA, 2006.

[16] G. Cormode, C. Procopiuc, D. Srivastava, E. Shen, and T. Yu, "Differentially private spatial decompositions," in *2012 IEEE 28th International Conference on Data Engineering*, pp. 20–31, New York, USA, 2012.

[17] F. D. McSherry, "Privacy integrated queries: an extensible platform for privacy-preserving data analysis," in *Proceedings of the 2009 ACM SIGMOD International Conference on Management of data*, pp. 19–30, Providence, Rhode Island, USA, 2009.

[18] K. G. Manton, *National Long-Term Care Survey: 1982, 1984, 1989, 1994, 1999, and 2004*, Inter-university Consortium for Political and Social Research, 2010.

[19] S. Ruggles, K. Genadek, R. Goeken, J. Grover, and M. Sobek, *Integrated public use microdata series: version 6.0 [dataset]*, vol. 23, Minneapolis: University of Minnesota, 2015.

[20] B. Tomhttp://fimi.uantwerpen.be/data/.