MGD: A Utility Metric for Private Data Publication

Zitao Li
li2490@purdue.edu
Purdue University

Trung Dang
dang28@purdue.edu
Purdue University

Tianhao Wang*
tianhao@virginia.edu
Carnegie Mellon University
& University of Virginia

Ninghui Li
ninghui@purdue.edu
Purdue University

ABSTRACT

Differential privacy has been accepted as one of the most popular techniques to protect user data privacy. A common way for utilizing private data under DP is to take an input dataset and synthesize a new dataset that preserves features of the input dataset while satisfying DP. A trade-off always exists between the strength of privacy protection and the utility of the final output: stronger privacy protection requires larger randomness, so the outputs usually have a larger variance and can be far from optimal. In this paper, we summarize our proposed metric for the NIST "A Better Meter Stick for Differential Privacy" competition [26], MarGinal Difference (MGD), for measuring the utility of a synthesized dataset. Our metric is based on earth mover distance. We introduce new features in our metric so that it is not affected by some small random noise that is unavoidable in the DP context but focuses more on the significant difference. We show that our metric can reflect the range query error better compared with other existing metrics. We introduce an efficient computation method based on the min-cost flow to alleviate the high computation cost of the earth mover's distance.

CCS CONCEPTS

• **Security and privacy → Privacy protections**.

KEYWORDS

Privacy, data synthesis, metric

ACM Reference Format:
Zitao Li, Trung Dang, Tianhao Wang, and Ninghui Li. 2021. MGD: A Utility Metric for Private Data Publication. In *8th International Conference on Networking, Systems and Security (8th NSysS 2021), December 21–23, 2021, Cox's Bazar, Bangladesh.* ACM, New York, NY, USA, 14 pages. https://doi.org/10.1145/3491371.3491385

1 INTRODUCTION

Driven by the successive user privacy regulations, such as GDPR [29] and CCPA [21], companies and governments have been putting more attention on how to perform data analysis without breaching user privacy. Differential privacy (DP) [9], as a *de facto* notion for privacy protection, has been applied on a large number of applications. Uber and LinkedIn build their differentially private database applications Flex [18], and Pinot [28] for data analysis. Google and Microsoft apply (local) differentially private mechanisms [8, 10, 12] to collect data from user devices. There are also a large number of efforts in the machine learning area to ensure the output model is private [1, 31], and the DP training has been integrated into both TensorFlow [14] and PyTorch [11]. Recently, DP has been applied by the US census bureau when the 2020 census statistics are published [2].

All the applications mentioned above are private mechanisms designed for specific tasks given user datasets. One disadvantage of these mechanisms is that privacy loss, which can be understood as the risk of privacy leakage, will be accumulated when the mechanisms are executed multiple times on the same dataset. One promising idea is to generate synthetic datasets with DP guarantee based on the real datasets to avoid privacy loss. Any mechanism executed on the differentially private synthetic dataset can be considered post-process and does not introduce additional privacy loss [9].

Many impressive differential private data synthesis methods have been proposed. Some of the methods are based on probabilistic graphical models. PrivBayes [32] synthesize data using a Bayesian network. The mechanism first privately determines the network structure, then obtains noisy marginals for the conditional probability distribution of each node to determine the probabilities of the edges. Another approach, PGM, which uses Markov Random Fields, was proposed in [24]. Both PrivBayes and PGM synthesize data by sampling from the privatized graphical model. Another more recent work, PrivSyn [33] first privately selects important marginals and then builds dense graph models based on those privatized marginals. When generating the synthetic data, it will first initialize a random dataset and update it until it matches the marginals.

Given the differentially private synthetic datasets generated by the above methods, how to evaluate the utility of the synthetic dataset can be tricky. In the DP data synthesis papers [32, 33], the ℓ_1 distance between the marginal tables of the ground truth dataset and the ones of synthetic data is a common metric. However, this metric can not capture the semantic meaning of the values. Another common metric used in the papers is to train classification models on both the ground truth dataset and the synthetic dataset, then compare the test accuracies of these two classification models on a ground truth test dataset. If these two models have similar test accuracies, then the synthetic dataset arguably has similar utility

*Work done at Purdue University.

8th NSysS 2021, December 21–23, 2021, Cox's Bazar, Bangladesh

ACM ISBN 978-1-4503-8737-8/21/12.
https://doi.org/10.1145/3491371.3491385

as the ground truth one. However, most machine learning models are not explainable, so it may be hard to interpret and quantify the utility. A third metric that appears in the literature is called range query. It generates random rang queries on some specified attributes and compares the results from the ground truth dataset and the synthetic dataset. The smaller the differences are, the closer these two datasets are. Compared with the marginal table distance, the range query difference also considers the basic semantic meaning of the value. However, it is only mainly used for numerical or ordinal datasets.

In this paper, we propose a metric, MarGinal Difference (MGD). The MGD consists of a set of Approximate Earth Mover Cost (AEMC) between the ground truth marginals and the marginals of the synthetic dataset. The AEMC is derived from the earth mover distance, so the MGD takes the semantics of the values into account. Our proposed metric is also configurable so that MGD can handle different kinds of attributes, whether numerical or categorical. Users can further specify the weights of different marginals so that the final metric score is proportional to the importance of the marginals. Because DP protects users' privacy with randomness, the generated datasets cannot be exactly the same as the ground truth dataset. Our metric ignores the small mismatches and only focuses on the significant differences.

The standard way to obtain an earth mover's distance is to solve an optimization problem. The computation cost of solving such an optimization problem is one of the major concerns for the earth mover's distance. We design an efficient algorithm for the AEMC based on the min-flow algorithm to ease the concern.

To summarize, our main contribution in this paper are the following:

- We propose a new utility metric for the differential private data publication. Our metric is configurable and can handle a mixture of numerical and categorical attributes. The semantic meaning of the values and the stochastic nature of the DP mechanisms are both considered by our metric.
- We show how to apply our metric to some real-world use cases and give a concrete case study to show the effectiveness of our metric.
- We design an efficient algorithm for our metric so that the computation time can be reduced significantly compared with the general solution with optimization packages.

Road map. In Section 2, we revisit the definition of DP and a private data synthesis methods. We also summarize the common metrics used in the existing literature for measuring the utility of the private dataset. In Section 3, we provide a high-level summary for our proposed metric. We formalized our metric in Section 4 and show use cases and a concrete case study in Section 5. We discuss how to alleviate computation cost in Section 6, and conclude with discussions in Section 7.

2 BACKGROUND

2.1 Differential privacy

Differential privacy was first introduced in [9]. Its secure setting assumes a central data curator has access to all users' data and publishes outputs of some mechanisms. Intuitively, DP requires that adding or removing any individual's record should have a limited impact on the mechanism's output.

Definition 1 ((ϵ, δ)-differential privacy). *A randomized mechanism $\mathcal{A}$ is differentially private iff given any pair of neighboring datasets $\mathbf{X}$ and $\mathbf{X}'$, the following holds for all possible output o:* $\Pr[\mathcal{A}(\mathbf{X}) = o] \leq e^{\epsilon} \Pr[\mathcal{A}(\mathbf{X}') = o] + \delta$.

The neighboring datasets are defined as any pairs of datasets that differ at exactly one user's record.

Private data synthesis. In this paper, we focus on designing a utility metric for synthetic datasets generated differential private mechanisms. Compared to designing specific algorithms for each task, such differential private data synthesis has the advantage that additional data analysis tasks can be performed without any further privacy concern: a property of differential privacy called post-processing allow us to arbitrarily process data that is already differentially private. Furthermore, existing algorithms for performing data analysis do not need to be modified.

The most promising existing method for the private generation of synthetic datasets uses probabilistic graphical models. The earliest work is PrivBayes [32], which uses a Bayesian network. It comprises two phases: (1) privately determines the network structure, and (2) obtains noisy marginals used as the basis for sampling along with the network. In 2018, NIST hosted a Differential Privacy Synthetic Data Challenge [25]. Three lines of methods are developed during and after this challenge.

- PGM [24] proposes a method only for sampling synthetic datasets (assuming the structure is chosen) by replacing the Bayesian network with Markov Random Fields. PGM (and PrivBayes) can only handle sparse fields. The authors then extend it to handle dense fields [23].
- PrivMRF developed a method only for privately choosing marginals based on Markov Random Fields [6]. The sampling procedure uses PGM.
- PrivSyn [33] is a complete toolkit that first privately selects important marginals and then builds dense graph models based on those privatized marginals. Unlike PrivBayes and PGM, which synthesize data by sampling from the privatized graphical model, PrivSyn first initializes a random dataset and updates it until it matches the marginals. Because of this, PrivSyn can naturally handle dense graphical models.

There are also other approaches that do not use probabilistic graphical models, such as [5, 13, 17], but they either are computationally inefficient or have poor empirical performance.

2.2 Existing common utility metrics

The following are some common metrics have been used to show the utility of the private synthetic dataset.

Marginal table distance metric. This metric computes the L_p distance between pairs k-way marginal tables, each pair having one from the ground truth dataset and the other from the synthetic dataset. The limitation of this metric is that it does not consider the semantic meaning of the values. For example, assume a 1-way marginal has 3 bins corresponding to frequencies of numerical values in intervals $[0, 10]$, $(10, 20]$ and $(20, 30]$. If the ground truth marginal is a vector $[5, 4, 1]$, and synthetic dataset A has marginal

[4, 5, 1] and synthetic dataset B has [4, 4, 2]. Both marginals from synthetic datasets have L_1 distance 2 to the ground truth marginal. However, synthetic dataset A is arguably close to the ground truth because the (10, 20] bin is semantically closer to [0, 10] than (20, 30].

Model performance metric. This metric evaluates the utility of the synthesis data based on the accuracy of the machine learning models. Classification machine learning models are trained on the ground truth and synthetic datasets. The closer the testing accuracy of the model trained on the synthetic dataset is to the testing accuracy of one trained on the ground truth dataset, the higher utility the synthetic dataset has maintained.

One of the motivations of this metric is that the performance of the classification model depends on the correlation between the attributes. If a model trained on a synthetic dataset has similar performance, then the correlations between the attributes are maintained well in the synthetic data. However, it is hard to quantitatively explain how the performance of the models can be related to the utility of the synthetic dataset. An accuracy drop of about 5% may be unacceptable on some datasets, but it may be considered a severe performance regression on other datasets. Besides, if we want to investigate whether some attributes are synthesized well while the others are not, comparing the results of may not explain the reason.

Range query metric. A third metric that has been widely used for DP mechanisms is the range query. A range query returns the number of records with values that fall in a specified range. When used as a metric in the DP literature, a large number (usually > 1000) of range queries with random ranges of some random attributes are applied on both the ground truth dataset and the synthetic dataset. The metric score is the average difference between the results on the ground truth dataset and the synthetic dataset.

One advantage of the range query metric is that range queries return the local frequencies to measure how the synthetic dataset maintains semantic meaning. However, range queries are not designed for categorical attributes, so there is no standard way to apply range queries when a dataset has both ordinal and categorical attributes.

Pie-chart metric. The pie-chart metric is used in Sprint 1 of the Differential Privacy Temporal Map Challenge [27]. It is a metric applied to the marginal table of attributes neighborhood, month and incident type. To remove the effect of the random noise on insignificant counts, the metric first set the counts less than 5% of the overall incidents count to zero. After the preprocessing, the penalty of the score consists of three parts: the Jensen–Shannon distance (JSD), the misleading presence penalty, and a bias penalty. The JSD mainly considers the over distribution, the second one penalizes the "false frequent" counts, and the third penalizes the seriously wrong total count. The final score is calculated as one minus the normalized penalty.

Compared with the previous ones, the improvement of this metric is that it disregards the effect of DP on the small incident counts and focuses on the accuracy of the significant incident counts. However, it is only motivated by and defined on a special use case. Another limitation of this metric, similar to the marginal table distance metric, is that it does not consider the semantic meaning of the attributes.

3 MGD METRIC OVERVIEW

We propose MarGinal Difference (MGD), a utility metric for private data publication of relational datasets. MGD assigns a difference score between a pair of datasets $\langle D_S, D_T \rangle$, where D_S is the synthesized dataset, and D_T is the ground truth. The high-level idea behind MGD is to measure the differences between many pairs marginal tables, each pair having one computed from D_S and one from D_T.

3.1 Design Desiderata

To turn the high-level idea of measuring differences between marginal tables into a concrete metric, we need to specify the following: which marginal tables are used in computing the score, how to measure the differences between two marginal tables, and how to combine these measurements into one score. We use the following design objectives to guide our choices for fleshing out the details of MGD.

- Flexibility. The set of marginal tables and their weights in deciding the final score can be configured. Under DP constraints, one cannot preserve all distribution information in the input dataset. Fortunately, it is often unnecessary for a synthetic dataset to preserver all distributions for it to be useful. Oftentimes not all distributions are equally important; however, the relative importance among different distributions, when they exist, cannot be determined without considering the attributes in the datasets and the application domain.
- Balancing Flexibility with Ease of Use. To provide flexibility, a metric needs to have configurable parameters; however, this may make a metric difficult to use. We solve this problem in two ways. First, instead of making everything configurable, we fix some aspects of the metric when we believe doing so preserves sufficient flexibility. Second, for almost all parameters, we provide default values that we hope would work for most application scenarios.
- Accommodation of Numerical Attributes. For attributes with numerical values, we want to take into consideration of the natural semantic distance between values. For example, for an age attribute, 19 is not very different from 20, but is very different from 90, and treating them just as three different values loses this semantic meaning.
- Accommodation of Structured Attributes. For many categorical attributes, the semantic meanings also suggest natural semantic distances between values. The metric should be able to use these semantic meanings when the application domain can benefit from doing so.
- Awareness of Noises. Under DP the data are noisy, and it is understood that small differences cannot be relied upon. The metric should reflect this, avoiding being dominated by many small differences.

3.2 Key Elements of MGD

The MGD metric is parameterized by the following.

- Data Schema. This specifies what attributes are in the dataset, the domain of possible values for each attribute, and any relationships between the values. At the minimum,

it should be clear whether each attribute is ordinal or not and the set of values that can be taken for each attribute. Optionally, one can also provide generalization hierarchies for these attributes.
- Target Marginal Schemas and Weights. This includes a set of marginal schemas where each schema specifies the attributes to be included. One applies each schema to D_S and D_T to obtain marginal tables and computes the AEMC score between the two resulting marginal tables. The final score is the weighted average of these AEMC scores, where the weights are equal by default but can be configured if one wants to.

Approximate Earth Mover Cost (AEMC) between Two Marginals. For measuring differences between two marginals, commonly used metrics include L_1, L_2, KS divergence, Jensen-Shannon distance, etc.. However, these metrics do not take into consideration of the underlying semantic distances between the values. The desire to use semantic distance values naturally led us to the earth mover's distance (EMD). Informally, suppose two distributions are interpreted as two different ways of piling up a certain amount of dirt over the region. In that case, the EMD is the minimum cost of turning one pile into the other, where the cost is assumed to be the amount of moving dirt times the distance by which it is moved.

We adapt EMD in three ways to suit our purpose and call the result AEMC. Instead of normalizing the marginal tables into probability distributions (i.e., all entries sum up to 1), we first use the unnormalized record counts to compute a score so that the absolute number of records also conveys information. The AEMC is obtained by dividing this score by the total number of records in the ground truth dataset. Second, in addition to moving counts from one bin to another, we also allow adding/removing counts directly. This helps deal with unequal total counts between the two datasets and make the metric a generalization of the L_1 distance between two marginal tables. Third, the earth moving does not need to get the two marginal tables exactly the same, but only so that the difference is below a threshold Δ. The small error tolerance avoids penalizing small differences in the counts. Without this feature, the score between two marginal tables can be dominated by cells that have small values when there are many such cells.

AEMC can be computed either by formulating it as a linear program and using a general linear programming solver or by formulating it as a flow problem in graphs and adapting existing algorithms to compute it. We have implemented both approaches and provide the details in the Appendix and code in github [1]. Experiments show that modeling AEMC as a flow problem results in a fast computation for fairly large marginals.

4 DEFINITION OF THE MGD METRIC

4.1 Technical Background

Earth Mover Distance (EMD). In statistics, EMD measures the distances between two probability distributions. EMD has been considered as an useful tool in computer vision area for the tasks including image retrieval [7], feature matching [15], and face verification [30]. In recent years, EMD is also widely used in deep learning models, such as generative adversarial network (GAN) [4, 16, 19]. EMD is also used in document similarity analysis [20] on high dimensional and sparse bag-of-words vectors.

More formally, EMD takes P and Q, each being a size T vector of non-negative values such that the sum over each vector is 1. Let $\mathbf{M}$ be a matrix of size $T \times T$ in which $\mathbf{M}_{ij} \geq 0$ is the cost of moving an element from the i^{th} position to the j^{th} position ($\mathbf{M}_{ii} = 0$ for all i values). The standard EMD is defined as:

$$\min_{\mathbf{X}} \sum_{i,j} \mathbf{X}_{ij}\mathbf{M}_{ij}$$
$$\text{s.t. } \forall i, \quad \sum_{j} \mathbf{X}_{ij} = P_i$$
$$\forall j, \quad \sum_{i} \mathbf{X}_{ij} = Q_j, \quad \forall i, j, \mathbf{X}_{ij} \geq 0$$

Intuitively, $\mathbf{X}_{ij}$ represents the amount one moves from the i^{th} element to the j^{th} element. EMD can be easily extended to vectors that are not probability distributions. There are also extensions to EMD that accommodate the situation where the sum of elements in P and the sum of elements in Q are different, either by allowing free disposal of excess quantities, or introducing cost for adding/removing values. We use the latter approach in our proposal.

Linear Programs (LP). Linear programming is used to optimize linear objective functions subject to linear equality and linear inequality constraints. One of the most canonical applications of linear programming is solving the network maximum flow problem. Also, it can be applied to regression problems and linear classification problems. Computing EMD is a special case of the linear programming problem. There exist software packages that can solve large LP instances reasonably fast.

Flow Algorithms. EMD can be computed by solving an instance of the transportation problem using any algorithm for minimum cost flow problem, e.g. the network simplex algorithm. Since such algorithms exploit the specific nature of the transportation problem, they are faster than using LP solvers for the same problem size.

Additional Resources. Any software package used in linear programming problems can calculate EMD and the new metric we proposed in this document. There are both open-sourced and commercial packages. With Python, there are CVXOPT and CVXPY, both of which are open-source libraries that can be used to solve linear programming problems. CPLEX, a commercial package developed by IBM, is another choice to solve the optimization problem. To accelerate the computation of AEMC, we reduce the problem to a special kind of min-cost problem and use OR-Tools solver (ortools python package) to solve the problem.

4.2 Formal Definition of AEMC

AEMC extends EMD in a few ways to be more suitable for our purpose. More formally, AEMC is parameterized by two parameters: (1) $\mathbf{M}$ is a $T \times T$ matrix where each cell $\mathbf{M}_{ij}$ is the distance between the i-th bin and the j-th bin. It is required that $\mathbf{M}_{ij} \in [0, 1] \cup \{\infty\}$ and $\forall_i\ \mathbf{M}_{ii} = 0$. Setting $\mathbf{M}_{ij} = \infty$ means prohibiting moving from bin i

[1] https://github.com/SkinfaxiL/NIST_metric

to bin j. Furthermore, (2) $\Delta > 0$ is a threshold such that differences between Δ are tolerated. AEMC is applied to a pair of size T marginals P and Q. Each element in P and Q corresponds to a count in the marginals. We view P and Q as two vectorized/flattened marginal tables of size $T \times 1$. P denotes the marginal computed from the synthesized dataset, and Q is computed from the ground truth. AEMC is defined as:

$$\begin{aligned}&\mathtt{AEMC}_{\mathbf{M},\Delta}(P,Q)\\&= \frac{1}{||Q||_1}\min_{\mathbf{X}} \sum_{i,j}\mathbf{X}_{ij}\mathbf{M}_{ij} + \sum_j \max\left\{\left|\sum_i \mathbf{X}_{ij} - Q_j\right| - \Delta, 0\right\}\\&\text{s.t. } \forall i, \quad \sum_j \mathbf{X}_{ij} = P_i \end{aligned} \tag{1}$$

Here $||Q||_1$ is the L_1 norm of Q, i.e., the sum of all components in the vector Q. The factor $\frac{1}{||Q||_1}$ normalizes the score so that it is independent of the total number of records in the dataset. Intuitively, each movement scheme is specified by a $T \times T$ matrix $\mathbf{X}$, such that $\mathbf{X}_{ij}$ gives the amount of moving from bin i to bin j. The condition $\sum_j \mathbf{X}_{ij} = P_i$ says that the total amount of quantity moving from bin i (some of which can be to itself) is exactly P_i. We want to minimize the cost, which has two components. The first one, $\sum_{i,j}\mathbf{X}_{ij}\mathbf{M}_{ij}$, is the cost of moving. Here we define $0 \cdot \infty$ to be 0, so when $\mathbf{M}_{ij} = \infty$, the corresponding $\mathbf{X}_{ij}$ should be 0 to minimize the above formula. The second one, $\sum_j \max\left\{\left|\sum_i \mathbf{X}_{ij} - Q_j\right| - \Delta, 0\right\}$, computes the penalty where the result after moving still differs from the ground truth Q. When the difference is below a threshold Δ, the penalty stays at 0. However, when the difference is above Δ, the part above Δ is penalized, representing the cost of adding/removing counts.

When $\mathbf{M}_{ij} = \infty$ for all $i \neq j$, and $\Delta = 0$, $\mathtt{AEMC}_{\mathbf{M},\Delta}$ approximates the L_1 distance. When P and Q are the same, or their L_∞ distance is $\leq \Delta$, the AEMC between them is 0.

4.3 Parameters and Configurations for MGD

The MGD metric has the following parameters. While many parameters can be configured, we list the default choices for almost all of them to simplify practical use of MGD.

4.3.1 Data Schema. Any dataset needs accompanying data schemas for it to make sense. To apply MGD, we require that the following information be extracted from the data schema for each attribute.

- *IsOrdinal*: whether the attribute is ordinal or not, i.e., whether there is a linear ordering among all values.
- *Domain D*: the set of all possible values for this attribute.
- *Generalization Hierarchy H*: a rooted tree such that:
  - There is a one-to-one correspondence between the leaves and each value in the domain D.
  - All leaves are at the same depth (distance from the root).
  - If an attribute is ordinal, the nodes at each level should appear (from left to right) in the order of small to large.

Whether an attribute is ordinal or not and its domain are usually determined by the nature of the dataset. We use a default hierarchy with two levels when an attribute does not have an explicitly specified generalization hierarchy. All values in the domain are the leaves, and there is one single root value. Allowing generalization hierarchy serves several purposes. First, they can be used to capture semantic distances among values in non-ordinal attributes. Second, they allow one to use marginals that involve attributes with many values while keeping the marginal size feasible. One can use generalized values for the attributes that have large domains. Third, for similar reasons as above, they enable marginals that involve more attributes.

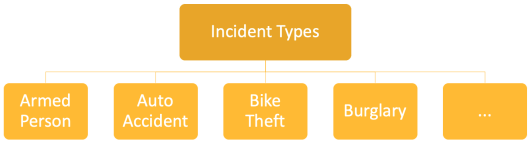
Semantic Distance. Given a generalization hierarchy, we define the level of a node as the number of edges from the node to the root. Thus the root has level 0, and the root's children have level 1, and so on. Whether an attribute is ordinal or not and its generalization hierarchy together fully determine the *Semantic Distance function* sd, which assigns a real value between 0 and 1 to *each pair of nodes in the same level*. For an ordinal attribute, when a level has $k > 1$ values, the distance between the i-th and the j-th value at that level is $\frac{|j-i|}{k-1}$. For two values of a non-ordinal attribute at level t, let s be the level of the lowest common ancestor of the two values, then the distance between the two values is defined to be $\frac{t-s}{t}$. Note that for any attribute in a marginal, only values from one specific level can appear, and we thus only need semantic distances between values at the same level. The definition normalizes the distances to be in $[0, 1]$ no matter which level is used.

Figure 1 shows two categorical attribute examples in generalization hierarchy structure. Figure 1(a) shows a default generalization hierarchy that has only two levels. The semantic distances between any two values, for example sd("Armed Person", "Bike Theft"), are all 1. Figure 1(b) shows the case that a categorical attribute has a customized hierarchy. Thus, the distance between any cities in the same state is $\frac{1}{2}$; the distance between any cities in different states and the distance between two states are both 1.

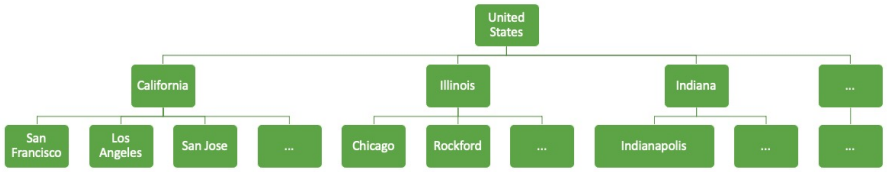
Figure 2 shows two ordinal attribute examples in generalization hierarchy structure. For Figure 2(a), where months are ordinal values, the semantic distance between any two months is the ordinal difference between them divided by 11. Figure 2(b) shows an ordinal attribute hierarchy with more than two levels. In this case, the semantic distance is defined for nodes in the same level. sd("1^{st} grade", "8^{th} grade") = $\frac{8-1}{13-1} = \frac{7}{12}$ because there are 13 values in leave level and the distance between those two values are 7; sd("Elementary school", "Middle school") = $\frac{2-1}{4-1} = \frac{1}{3}$ because there are only 4 values in the second level.

4.3.2 Target Marginals. MGD requires the specification of a set Φ of target marginals, each $\phi \in \Phi$ consisting of the following:

- Attributes in the Marginal. Users need to specify a set of attributes included in the marginal, and for each selected attribute, which level of the generalization hierarchy is used to compute the marginal. The default level for each selected attribute is the leaf level. All the bins in the target marginal are determined after the attributes are specified. Alternatively, this can be defined as selecting a generalization level for every attribute. If the root level is selected for an attribute, all records have the same value, and that attribute is effectively not included in the marginal.

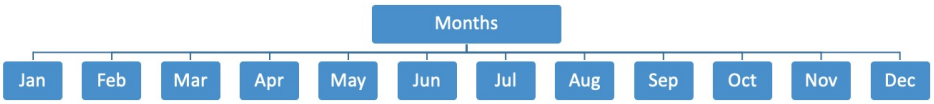


(a) Generalization hierarchy of attribute "incident type" in 2019 Baltimore 911-Call and Police Incident data.

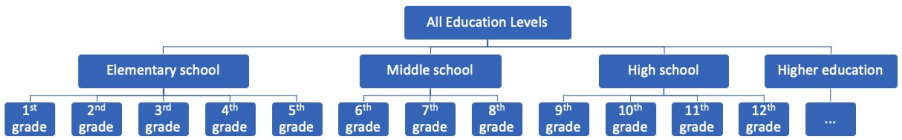


(b) Generalization hierarchy of cities in United States.

Figure 1: Generalization hierarchy of categorical attributes.



(a) Generalization hierarchy of attribute "month" in 2019 Baltimore 911-Call and Police Incident data.



(b) Generalization hierarchy of education levels in United States.

Figure 2: Generalization hierarchy of categorical attributes.

- Attribute Weights. A weight w_a for each selected attribute a. The weight $w_a \in \{\infty\} \cup [0, 1]$, and if any attribute has a weight in $[0, 1]$, such weights for all attributes sum up to 1. If an attribute is assigned a weight of ∞, it means that when computing the difference score on this marginal, one does not want to consider moving between bins where this attribute has different values.
  The assigned weights and the semantic distance function sd together define the bin distance matrix $\mathbf{M}^{\phi}$ needed for computing the AEMC. Given two bins b and b' in the marginal, if there exists an attribute such that b and b' differ in an attribute that has a weight of ∞, the distance between b and b' is ∞. Otherwise, the bin distance:

$$\mathbf{M}^{\phi}_{b,b'} = \sum_{a} w_a \cdot \mathrm{sd}(b_a, b'_a) \tag{2}$$

  where b_a denotes value for attribute a of the bin b. The matrix $\mathbf{M}^{\phi}$ is used in Equation (1) to calculate AEMC. Note that the distance between two cells are either ∞ or a value between $[0, 1]$. When the distance is ∞, this means that this marginal one does not want to move between the two bins in computing the earth mover distance.
  The default for weight assignment is to assign the weight of ∞ to each categorical attribute and equal weight to each numerical attribute.
- Tolerance threshold Δ^{ϕ}. The threshold under which differences in bins are not penalized in AEMC. (See Equation (1)).
- Marginal Score Weight ω^{ϕ}. It is a positive number. The weight specifies how much the AEMC computed on the marginal ϕ contributes to the final MGD score.

Summary of Difference Measures in defining MGD . There are several notions that measure differences between two things here. For clarity, we recap them here.

- Semantic Distance sd is for the distance between two values for the same attribute. It is defined for any pair of values that are on the same level of the generalization hierarchy, and is fully determined by the generalization hierarchy and whether the attribute is ordinal or not. Its value is always in $[0, 1]$.
- Bin Distance $\mathbf{M}^{\phi}$, defined in Equation (2), uses the semantic distance to define the distance between two bins in one marginal. Each bin is specified by the value that each attribute can take. $\mathbf{M}^{\phi}$ is defined as the weighted sum of semantic distances on all attributes. It is a generalization of the Manhattan distance by adding a weight to each dimension. We choose to use a generalization of Manhattan distance (instead of, e.g., Euclidean distance) in part because this enables faster computation of AEMC.
- AEMC, defined in Equation (1), is for measuring the difference between a pair of marginal tables, one computed from the synthesized dataset, and the other from the ground truth dataset.
- MGD score is the weighted average of the AEMC score for all marginals in Φ. More specifically:

$$\mathtt{MGD}_{\Phi}(D_S, D_T) = \left(\sum_{\phi \in \Phi} \omega^{\phi} \cdot \mathtt{AEMC}_{\mathbf{M}^{\phi}, \Delta^{\phi}}\left(\phi(D_S), \phi(D_T)\right)\right) / \left(\sum_{\phi \in \Phi} \omega^{\phi}\right) \tag{3}$$

where $\phi(D)$ is the marginal table obtained from dataset D using the marginal schema ϕ and flattened to a vector.

4.4 More Fine-grained Utility Information

MGD provides a single final score. However, since MGD is based on the AEMC scores for different marginals, examining these AEMC scores provides more fine-grained utility information. One can see which marginals are preserved well and which are not. It is also possible to expose more detailed information in computing AEMC to provide an even deeper dive. For example, within one marginal, we can compute the contribution of each attribute to the AEMC score, illustrating the information on which attribute is less accurately preserved. It is even possible to compute the contribution of individual bins to the AEMC score and output the ones that have the most contribution, illustrating where the errors concentrate.

5 APPLICATIONS OF THE MGD METRIC

Here we discuss applications of the MGD metric. We discuss an application of MGD first to census-style data used in NIST's 2018 Differential Privacy Synthetic Data Challenge [25], then to temporal map data in NIST's 2020 Differential Privacy Temporal Map Challenge [26]. Finally, we show experimental results using one concrete dataset from [27].

5.1 Application to Census-type Data

Here we demonstrate how to apply MGD to census data. As an application example, we consider the Public Use Microdata Sample (PUMS) of the 1940 USA Census Data used in Phase 3 of the 2018 Differential Privacy Synthetic Data challenge [25]. The dataset has 98 attributes. In the challenge, three metrics were used.

- Density Estimation. For this metric, the scoring algorithm randomly samples 300 marginal schemas, each with three randomly chosen attributes. Then, compute the normalized marginal tables from the synthetic dataset and the ground truth dataset for each marginal schema. Then, use the L_1 distance between the two marginal tables as the penalty. This can be defined in a straightforward way using MGD.
- Range Query. For this metric, the scoring algorithm randomly samples 300 range queries and assesses the synthetic dataset's accuracy to answer these queries. This cannot be directly implemented using MGD, although this score is highly correlated with the above density estimation score computed from 3-way marginals, as observed during the competition.
- Gini Index and Rank Accuracy. For each city present in the dataset (as specified by the CITY column), we calculate, based on the SEX and INCWAGE columns, the Gini index and gender pay gap. This utility can be captured by using the 3-way marginal over CITY, SEX, and INCWAGE. If a dataset more accurately preserves this 3-way marginal, it would also result in more accurate estimations of the Gini index and rank of cities based on the gender pay gap.

If one wants a MGD scheme that correlates highly with the scoring scheme used in this challenge (which involves the three metrics mentioned above), one can use a sufficiently large number of 3-way marginals, and ensuring that the marginal involving CITY, SEX, and INCWAGE has a high weight, while treating INCWAGE as an ordinal attribute.

5.2 Application to Temporal Map Data

Temporal map data are relational data where some important attributes represent time and space. When applying MGD to temporal map data, there are several considerations.

- The temporal spatial nature of the dataset can guide the choice of which marginals are included in Φ to compute the MGD score. For example, when there are 40 attributes, considering all 3-way marginals may be too many, and one may want to consider all 3-way marginals that include at least one of the temporal or spatial attributes.
- The temporal attribute is naturally ordinal. When temporal attributes are used in NIST competitions, they are often processed to have a limited set of values. For example, the Police Incident Data used in Spring 1 generalizes the time attribute to month. Similarly, the time attribute in the San Francisco Fire dataset, which spans several years, was bucketed to 100 values. In some sense, the ground truth dataset has already lost much information. This was because using finer-grained values results in sparse distribution, which previous metrics cannot handle well. MGD has several features that make it more capable of handling finer-grained ordinal attributes. First, the usage of semantic distance enables one to distinguish synthetic datasets that have the time attribute approximately but are not completely correct. Second, by defining a generalization hierarchy over the time attribute,

one has the flexibility of measuring both more precise distribution over the time attribute (e.g., by using a one-way marginal over the time), and correlation of time with other attributes (e.g., by using multi-way marginals in which more coarse-grained values for time are used).
- The spatial attributes are similar to the time attributes in that they can be ordinal. It is also possible to extend MGD to handle spatial attributes better. For example, for computational efficiency considerations, we use the generalization hierarchy to derive the semantic distance function. However, when there are natural distance measures for a spatial attribute, it is possible to specify a distance matrix that directly assigns a distance to any two values. The trade-off is that we can no longer exploit the hierarchical structure underlying the semantic distance function and have to explicitly consider movement between any pair of values, resulting in a quadratic number of movement variables.

5.3 A Concrete Case Study

We consider the dataset used in Sprint 1 of the Differential Privacy Temporal Map Challenge [27], namely Baltimore 911 Call and Police Incident Data. There sample dataset in the contest has three attributes: time (12 months), neighborhood (278 values), and incident type (174 values). Below, we show how different types of information one may be interested in the data can be represented with MGD.

- Per neighborhood-month distribution of incident types. The utility of property is what the pie-chart metric used in Sprint 1 tries to measure. Conceptually, the dataset is first partitioned into $12 \times 278 = 3336$ parts (one for each month and neighborhood), and for each part, the distribution over the 174 incident type is compared to the ground truth.
  To simulate this using MGD, one uses a 3-way marginal over time, neighborhood, and incident type, with all weights being ∞, and a suitable Δ, which can be a small constant. (We use $\Delta = 2$ in our experiments. Such a metric is similar to L_1 distance, but ignores the impact of small differences, which is in the same spirit as the pie-chart metric ignoring densities for incident types that are below 5%.
- Per neighborhood total incidents over time. Here we assume that one is only interested in how the total number of incidents (aggregating over all incident types) changes over time for each neighborhood.
- Distribution of certain types of incidents over neighborhoods. To determine whether adequate resources are available to handle incidents of certain types, we can consider a two-way marginal over neighborhood and incident type. When we want to analyze different incident types separately, we define the weight for incident type as ∞ so that counts from bins with different incident types cannot flow into each other. The overall score thus reflects accuracy within each incident type. One can also dive down into the details and look at movement cost for each incident type separately. Assuming that a meaningful distance measure exists for the neighborhood, one can run hierarchical clustering to develop a generalization hierarchy.

When one does not have a specific goal in mind, one could choose to measure the three one-way marginals, the three two-way marginals, and the three-way marginal. One possible marginal weight assignment is 1/9 for each one-way marginals, 1/9 for two-way marginals, and 1/3 for the three-way marginal.

In this paper, we apply MGD on the synthetic datasets generated by the following methods, with the police incident data as input.

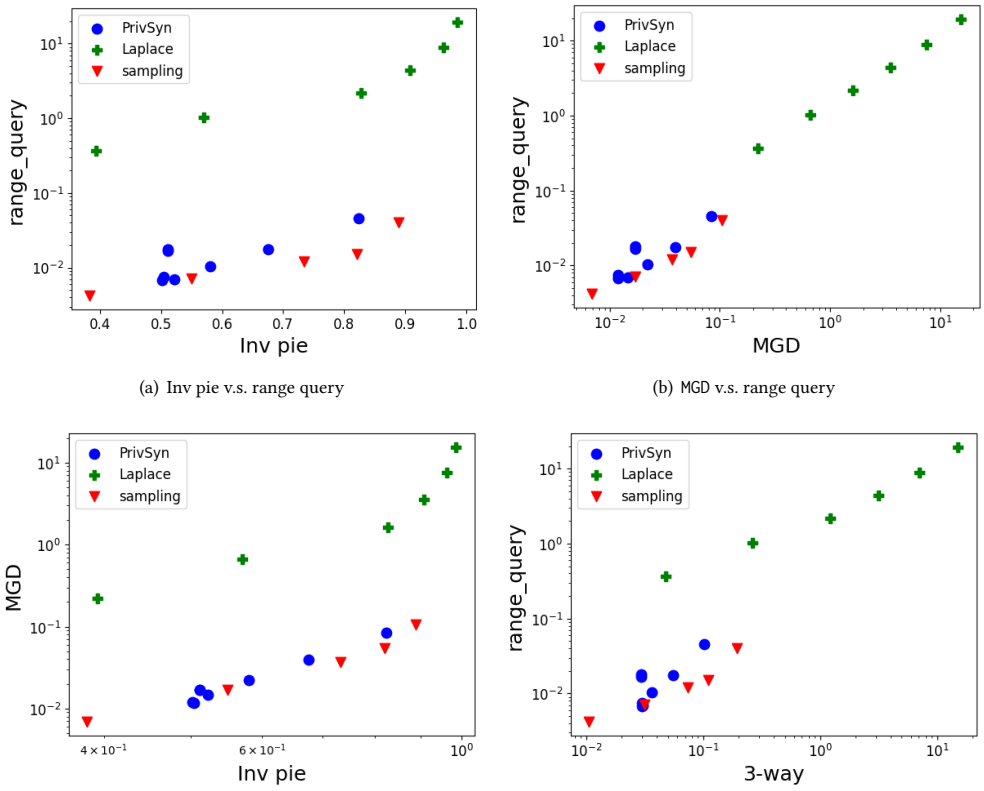
- PrivSyn: PrivSyn [33] is one of the state-of-the-art differential private data synthesis mechanisms. We generate eight synthetic datasets with PrivSyn and different privacy budgets $\epsilon \in \{0.1, 0.25, 0.5, 1.0, 2.0, 4.0, 10.0, 100.0\}$
- Laplace: The Laplace mechanism is considered as a benchmark in the NIST competition [27], but its synthetic datasets should have inferior performance compared to the one from PrivSyn. We generate six synthetic datasets with the Laplace mechanism and privacy budgets $\epsilon \in \{0.25, 0.5, 1.0, 2.0, 4.0, 10.0\}$. We use the Laplace mechanism to process the marginal and sampling from the marginal after that.
- Sampling: We also generate four datasets by sampling from the ground truth dataset with sampling probability $\{0.01, 0.05, 0.1, 0.5\}$. These four datasets are not private but at different representative levels of the ground truth dataset. Thus, they can be used as benchmarks to show how the metric scores change with different similarity levels.

Figure 3 shows four scores from different metrics: (1) average error for randomly generated range queries; (2) Inv-Pie (short for Inverse Pie Chart), defined to be $(1-S/3336)$, where S is the pie-chart score of the dataset and 3336 is the total number of neighborhood-month pairs; (3) the MGD score using the above weighting scheme (i.e., 1/9 for each of 1-way and 2-way marginals, and 1/3 for the 3-way marginal); and (4) the AEMC scores for the 3-way marginal, which can also be viewed as MGD scores that use only the 3-way marginal. Because the range for (1), (3), and (4) is large, they are plotted on a log scale. Each point in Figure 3 represents one dataset, and a few different methods are used to generate synthetic data. See Appendix B for details.

From Figure 3, we can see that MGD correlates well with the range query error. AEMC correlates worse because it considers only the accuracy of individual cells of 3-way marginals and not accumulated errors over a range of values. The pie-chart scores do not correlate well with range query error because they were not designed to do so.

6 COMPUTATION ISSUES

Computing the MGD requires computing the AEMC between two marginals. We now present two approaches for computing AEMC. The first approach, which is more general, is to transform the optimization problem of computing AEMC into a linear program and use the existing LP solver. The second approach, which is more efficient, is to model AEMC as a variant of the min-cost network flow problem.



(a) Inv pie v.s. range query

(b) MGD v.s. range query

(c) Inv pie v.s. AEMC (3-way)

(d) AEMC (3-way) v.s. range query

Figure 3: Comparison between range query, Inv Pie, MGD and AEMC (3-way). For MGD, we user weight $\frac{1}{3}$ for the 3-way marginal, ant $\frac{1}{9}$ for the other 6 marginals

.

6.1 Transform AEMC as a Linear Program

Recall that AEMC is defined as:

$$\text{AEMC}_{\mathbf{M},\Delta}(P,Q) = \min_{\mathbf{X}} \sum_{i,j} \mathbf{X}_{ij}\mathbf{M}_{ij} + \sum_j \max\left\{\left|\sum_i \mathbf{X}_{ij} - Q_j\right| - \Delta, 0\right\}$$

$$\text{s.t. } \forall i, \quad \sum_j \mathbf{X}_{ij} = P_i$$

The use of absolute values in the objective function of AEMC can be removed by introducing additional dummy variables. The above is equivalent to the following linear program:

$$\min_{\mathbf{X},\mathbf{e}} \sum_j e_j$$

$$\text{s.t. } \forall j, \quad e_j \geq \sum_i \mathbf{X}_{ij}\mathbf{M}_{ij} + \left(\sum_i \mathbf{X}_{ij} - Q_j - \Delta\right)$$

$$e_j \geq \sum_i \mathbf{X}_{ij}\mathbf{M}_{ij} + \left(-\sum_i \mathbf{X}_{ij} + Q_j - \Delta\right)$$

$$e_j \geq \sum_i \mathbf{X}_{ij}\mathbf{M}_{ij}$$

$$\forall i, \quad \sum_j \mathbf{X}_{ij} = P_i$$

With $\mathbf{M}, \Delta, P, Q$ as the input to an optimization solver, we can obtain the AEMC between two marginals represented by P and Q.

6.2 Model AEMC as Min-cost Flow Problem

The AEMC is equivalent to a min-cost flow problem with some edges requiring both maximum and minimum flow through them. High-level speaking, the customized min-cost flow problem asks for the minimum cost of flows from source node s to sink node t. Between s and t, there are two sets, each having T nodes. The nodes in the first set correspond to the bins of P, while the nodes in the other set correspond to Q. There are flows in the following scenarios:

(1) From s to all the nodes in first set. The flow from s to the i^{th} node in the first set is constrained to P_i, with cost 0.
(2) From s to all the nodes in the second set. The flow from s to the j^{th} node in the second set is constrained to be non-negative, with cost 1.
(3) From the nodes in the first set and the nodes in the second set. The flow from i^{th} node in the first set to the j^{th} node in the second set is constrained to be non-negative and have cost $\mathbf{M}_{ij}$.
(4) From the nodes in the second set to t. There are two flows from the j^{th} node for $j \in [T]$ in the second set to t. One is constrained to be $[Q_j - \Delta, Q_j + \Delta]$ and with cost 0. The other

flow between j^{th} node in the second set has non-negative constraints but cost 1.

We detailed the algorithm in Appendix A.

6.3 Optimization of AEMC

Generally speaking, there are $O(T^2)$ variables in the AEMC optimization problem, each of which corresponds to a movement of counts (flow) from bin b to bin b'. In the second approach, these $O(T^2)$ variables manifest as $O(T^2)$ edges. However, as introduced in [22], the number of variables for solving earth-mover distance (also applied to AEMC in our case) can be reduced to $O(T)$ if each bin b has a set of "neighbors" $\mathsf{N}(b)$ such that the number of neighbors for each bin is a constant independent from the domain size of any attribute, and all flows between any two bins b and b' can be replaced by a sequence of flows between only neighbors, with the same cost. That is,

$$\forall b, b', \exists b_1 \in \mathsf{N}(b), b_2 \in \mathsf{N}(b_1), \ldots, b' \in \mathsf{N}(b_k),$$
$$\mathbf{M}_{b,b'} = \mathbf{M}_{b,b_1} + \mathbf{M}_{b_1,b_2} + \ldots + \mathbf{M}_{b_k,b'}$$

Several design features of AEMC aim at enabling the above optimization, which we now discuss.

- The L_1 nature of bin distance. The way the bin distance matrix is defined, namely $\mathbf{M}_{bb'} = \sum_a w_a \cdot \mathrm{sd}(b_a, b'_a)$, means that we can limit neighboring bins for any bin only to those that differ in exactly one attribute.
- Ordinal Attributes. The way semantic distance is defined for ordinal attributes means that for each value v, one only needs to consider the value just above v and the value just below v as v's neighbors.
- Non-ordinal Attributes. Recall that we define the semantic distance between two values of a non-ordinal attribute at level t of the generalization hierarchy as $\frac{t-s}{t}$, where s is the level of the lowest common ancestor of the two values. Using this, we can introduce new dummy values for this attribute, with one value for each node at a level $< t$. Then, each value only neighbors by its parent and descendent. The cost of moving between a parent and child node has cost $\frac{1}{2\times t}$. Adding these new dummy values will create new dummy bins, and constraints need to be added to ensure that they start and end with 0 counts in the flow.

6.4 Scalability and Feasibility

We have found that the linear programming approach of computing AEMC scales to thousands of bins in marginals but have difficulty dealing with tens of thousands or more bins. On the other hand, modeling AEMC as a min-cost flow problem and solving the instance using OR-Tools (https://developers.google.com/optimization), we can compute the AEMC between the ground truth 3-way marginal of Police Incident Data (totally 580464 cells) and the privatized one in 2 minutes.

We suggest avoiding using marginals with too many cells, both for scalability concerns and effectiveness in measuring meaningful differences between marginals. In general, if most cells in the marginal table from the ground truth dataset have a very low count, not much meaningful information can be obtained under the constraint of DP.

6.5 Exploration of Parameter Tuning

Section 5.3 showed that depending on how one wants to use the final data, one can choose different weights for marginals. The generalization hierarchies will also depend on the application domain, and the nature of the attributes. Another parameter that one can choose is Δ. We currently recommend choosing Δ to be a small constant, e.g., some value between 2 and 10. Generally speaking, it would be meaningful to set a smaller Δ for a synthetic dataset with a large privacy budget ϵ and focus more on the utility; set larger Δ for those with small ϵ in order to tolerate the noise required by the strong privacy. Note that small differences have a larger impact when accumulated, which can be captured by using more coarse-grained marginals with fewer cells. In this paper, we mainly explore the usability of our metric based on the temporal map data in NIST competitions. Therefore, we do not yet have a deep understanding of the impact of the exact choice of Δ in other tasks. We mark this as a research direction for future study.

7 CONCLUSIONS AND DISCUSSIONS

We introduce the MGD metric, which can be configured to serve individual needs. We show how the MGD can capture the semantic meaning of attributes and handle various data types. We also introduce an efficient algorithm that reduces the computation significantly. One main challenge for designing a metric is that there is no obvious metric for assessing proposed metrics. There are many data analysis tasks one may be interested in, and one synthetic dataset may perform very well on one task but poorly on another. Thus no single metric can be simultaneously "accurate" for all tasks. Without fixing the set of data analysis tasks, we cannot think of a better metric than measuring the accuracy of many marginals. Measuring L_1 differences on either all 2-way and 3-way marginals (or a randomly selected subset of them if there are too many) is an excellent starting point, but it does not capture the semantic meaning of the attributes.

Our proposed MGD metric in this paper can be viewed as an attempt to improve the basic L_1 difference on marginals by addressing some limitations and issues we have experienced with the metric, both through our prior research and through our experiences in the NIST Differential Privacy Synthetic Data challenge. One goal is to exploit semantic meanings among the values, both ordinal ones, and non-ordinal ones. In the last NIST competition, we observed that when an attribute has a large domain, almost all marginals involving that attribute tend to have close to maximal L_1 error. This is because the vast majority of the bins have 0 or close to 0 count. After adding noises, which cells have non-zero counts are random. Thus even if a synthesized dataset can capture the few high counts of interest, the L_1 error would still be dominated by those caused by what are essentially white noises. We introduce the approximate nature to enable one to avoid this issue.

We believe that for almost any data analysis task, one can find a suitable subset of marginals and corresponding weight so that the MGD scores are highly correlated with accuracy in the data analysis task. However, the challenge in using MGD may be to identify the right configurations for the application.

Acknowledgement. This work is supported in part by the United States National Science Foundation under Grant No. 1931443.

REFERENCES

[1] Martín Abadi, Andy Chu, Ian Goodfellow, H Brendan McMahan, Ilya Mironov, Kunal Talwar, and Li Zhang. 2016. Deep learning with differential privacy. In *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*. ACM, 308–318.

[2] John M Abowd. 2018. The US Census Bureau adopts differential privacy. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 2867–2867.

[3] Ravindra K Ahuja, Thomas L Magnanti, and James B Orlin. 1993. *Network Flows: Theory, Algorithms, and Applications*. Prentice hall.

[4] Martin Arjovsky, Soumith Chintala, and Léon Bottou. 2017. Wasserstein generative adversarial networks. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*. 214–223.

[5] Avrim Blum, Katrina Ligett, and Aaron Roth. 2008. A learning theory approach to non-interactive database privacy. In *STOC*. 609–618.

[6] Kuntai Cai, Xiaoyu Lei, Jianxin Wei, and Xiaokui Xiao. 2021. Data Synthesis via Differentially Private Markov Random Fields. *Proceedings of the VLDB Endowment* 13 (2021).

[7] Scott Cohen and L Guibasm. 1999. The earth mover's distance under transformation sets. In *Proceedings of the Seventh IEEE International Conference on Computer Vision*, Vol. 2. IEEE, 1076–1083.

[8] Bolin Ding, Janardhan Kulkarni, and Sergey Yekhanin. 2017. Collecting Telemetry Data Privately. In *Advances in Neural Information Processing Systems*. 3574–3583.

[9] Cynthia Dwork, Frank McSherry, Kobbi Nissim, and Adam Smith. 2006. Calibrating Noise to Sensitivity in Private Data Analysis. In *TCC*. 265–284.

[10] Úlfar Erlingsson, Vasyl Pihur, and Aleksandra Korolova. 2014. Rappor: Randomized aggregatable privacy-preserving ordinal response. In *Proceedings of the 2014 ACM SIGSAC conference on computer and communications security*. ACM, 1054–1067.

[11] Facebook. [n.d.]. Opacus. https://opacus.ai/.

[12] Giulia Fanti, Vasyl Pihur, and Úlfar Erlingsson. 2016. Building a RAPPOR with the Unknown: Privacy-Preserving Learning of Associations and Data Dictionaries. *Proceedings on Privacy Enhancing Technologies (PoPETS)* issue 3, 2016 (2016).

[13] Marco Gaboardi, Emilio Jesús Gallego Arias, Justin Hsu, Aaron Roth, and Zhiwei Steven Wu. 2014. Dual Query: Practical private query release for high dimensional data. In *International Conference on Machine Learning*. 1170–1178.

[14] Google. [n.d.]. TensorFlow Privacy. https://github.com/tensorflow/privacy.

[15] Kristen Grauman and Trevor Darrell. 2004. Fast contour matching using approximate earth mover's distance. In *Proceedings of the 2004 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2004. CVPR 2004.*, Vol. 1. IEEE, I–I.

[16] Ishaan Gulrajani, Faruk Ahmed, Martin Arjovsky, Vincent Dumoulin, and Aaron C Courville. 2017. Improved training of wasserstein gans. In *Advances in neural information processing systems*. 5767–5777.

[17] Moritz Hardt, Katrina Ligett, and Frank McSherry. 2012. A simple and practical algorithm for differentially private data release. In *Advances in Neural Information Processing Systems*. 2339–2347.

[18] Noah Johnson, Joseph P Near, and Dawn Song. 2018. Towards practical differential privacy for SQL queries. *Proceedings of the VLDB Endowment* 11, 5 (2018), 526–539.

[19] Tero Karras, Timo Aila, Samuli Laine, and Jaakko Lehtinen. 2018. Progressive Growing of GANs for Improved Quality, Stability, and Variation. In *International Conference on Learning Representations*.

[20] Matt Kusner, Yu Sun, Nicholas Kolkin, and Kilian Weinberger. 2015. From word embeddings to document distances. In *International conference on machine learning*. 957–966.

[21] California State Legislature. [n.d.]. California Consumer Privacy Act of 2018. https://leginfo.legislature.ca.gov/faces/codes_displayText.xhtml?division=3.&part=4.&lawCode=CIV&title=1.81.5.

[22] Haibin Ling and Kazunori Okada. 2007. An efficient earth mover's distance algorithm for robust histogram comparison. *IEEE transactions on pattern analysis and machine intelligence* 29, 5 (2007), 840–853.

[23] Ryan McKenna, Siddhan Pradhan, Daniel Sheldon, and Gerome Miklau. 2021. Relaxed Marginal Consistency for Differentially Private Query Answering. *arXiv* (2021).

[24] Ryan McKenna, Daniel Sheldon, and Gerome Miklau. 2019. Graphical-model based estimation and inference for differential privacy. In *International Conference on Machine Learning*. PMLR, 4435–4444.

[25] NIST. [n.d.]. 2018 Differential Privacy Synthetic Data Challenge. https://www.nist.gov/ctl/pscr/open-innovation-prize-challenges/past-prize-challenges/2018-differential-privacy-synthetic.

[26] NIST. [n.d.]. DeID2 - A Better Meter Stick for Differential Privacy. https://www.herox.com/bettermeterstick/teams.

[27] NIST. [n.d.]. Differential Privacy Temporal Map Challenge: Sprint 1. https://www.drivendata.org/competitions/69/deid2-sprint-1-prescreened/page/263/.

[28] Ryan Rogers, Subbu Subramaniam, Sean Peng, David Durfee, Seunghyun Lee, Santosh Kumar Kancha, Shraddha Sahay, and Parvez Ahammad. 2020. LinkedIn's Audience Engagements API: A privacy preserving data analytics system at scale. *arXiv preprint arXiv:2002.05839* (2020).

[29] Paul Voigt and Axel Von dem Bussche. 2017. The eu general data protection regulation (gdpr). *A Practical Guide, 1st Ed., Cham: Springer International Publishing* 10 (2017), 3152676.

[30] Fan Wang and Leonidas J Guibas. 2012. Supervised earth mover's distance learning and its computer vision applications. In *European Conference on Computer Vision*. Springer, 442–455.

[31] Xi Wu, Fengan Li, Arun Kumar, Kamalika Chaudhuri, Somesh Jha, and Jeffrey F. Naughton. 2017. Bolt-on Differential Privacy for Scalable Stochastic Gradient Descent-based Analytics. In *Proceedings of the 2017 ACM International Conference on Management of Data (SIGMOD)*. 1307–1322.

[32] Jun Zhang, Graham Cormode, Cecilia M Procopiuc, Divesh Srivastava, and Xiaokui Xiao. 2017. Privbayes: Private data release via bayesian networks. *ACM Transactions on Database Systems (TODS)* 42, 4 (2017), 25.

[33] Zhikun Zhang, Tianhao Wang, Ninghui Li, Jean Honorio, Michael Backes, Shibo He, Jiming Chen, and Yang Zhang. 2021. Privsyn: Differentially private data synthesis. In *30th {USENIX} Security Symposium ({USENIX} Security 21)*.

A USING MIN-COST FLOW WITH LOWER BOUNDS TO COMPUTE AEMC

The *flow with lower bounds* problem is similar to the classic flow problem but with an additional constraint that there is a lower bound on each edge. More formally, given a graph $G = (V, E)$ with a source s and a sink t, where each edge (u, v) has a capacity $c(u, v)$ and a lower bound $b(u, v)$, the goal is to find a flow f from s to t such that

$$b(u,v) \le f(u,v) \le c(u,v) \qquad \forall (u,v) \in E$$
$$\sum_{(x,u)\in E} f(x,u) - \sum_{(u,y)\in E} f(u,y) = 0 \qquad \forall u \in V \setminus \{s,t\}$$

Such a flow is called a valid flow on G. The *min-cost flow with lower bounds* problem gives each edge (u, v) a cost $w(u, v)$, and we need to find a valid flow that minimizes

$$\sum_{(u,v)\in E} f(u,v) \cdot w(u,v)$$

We first show how to reduce the problem of computing AEMC to min-cost flow with lower bounds, and then show how this can be reduced to min-cost circulation problem, for which we use the standard algorithm implemented in software packages such as OR-Tools to solve.

A.1 Reducing Computing AEMC to Min-cost Flow with Lower Bounds

For an AEMC instance, we create a graph with 4 layers, where the first layer contains only the source node s, the second layer contains n nodes representing cells of the marginal P computed from the privatized data, the third layer contains n nodes representing cells of the marginal Q computed from the ground truth, and the fourth layer contains only the sink node. The Figure 4 below shows an example of $n = 3$ bins.

Let us first restrict our problem: suppose we can only move so that each bin q_i lies within the approximated range of Q_i, i.e. $Q_i - \Delta \le q_i \le Q_i + \Delta$.

We will add edges (s, p_i) with lower bound $b(s, p_i) = P_i$, capacity $c(s, p_i) = P_i$, and unit cost $w(s, p_i) = 0$ for all $1 \le i \le n$. Intuitively, think of these edges as "whether we want it or not, each p_i must have P_i value at first". Similarly, we will add edges (q_i, t) with lower bound $b(q_i, t) = Q_i - \Delta$, capacity $c(q_i, t) = Q_i + \Delta$, and unit

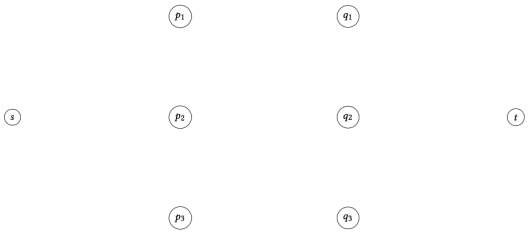


Figure 4: Initializing with $n = 3$ bins.

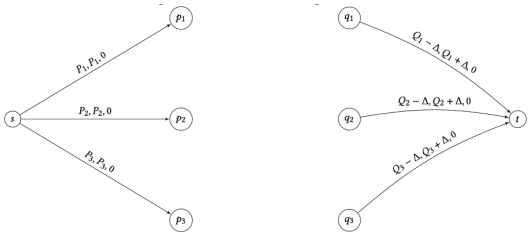


Figure 5: Adding restrictions $Q_i - \Delta \le q_i \le Q_i + \Delta$.

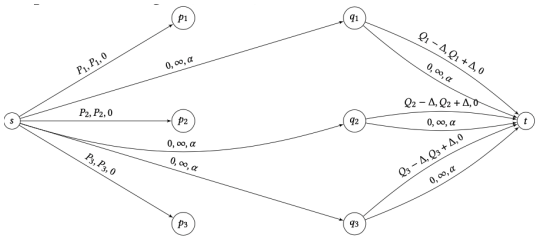


Figure 6: Allowing each resulting bin to exceed the predefined range with a unit cost of α.

cost $w(q_i, t) = 0$, with the intuition that each resulting bin must contribute at least $Q_i - \Delta$ and at most $Q_i + \Delta$ (as shown in Figure 5).

Let us relax the restriction and see how we can allow each resulting bin to exceed the predefined range with a unit cost of α. (In definition of AEMC, the value of α is implicitly set to 1. Here we describe the general case, when α can be set to other values). We can do it as follows as shown in Figure 6: for each node q_i, add an edge (s, q_i) with lower bound $b = 0$, capacity $c = \infty$, and unit cost $w = \alpha$; additionally, add an edge (q_i, t) with lower bound $b = 0$, capacity $c = \infty$, and unit cost $w = \alpha$. Intuitively, the edge (s, q_i) helps q_i to achieve the predefined range $[Q_i - \Delta, Q_i + \Delta]$ with a unit cost of α; similarly, the edge (q_i, t) helps q_i to "relieve" some flow to allow it to reach the predefined range $[Q_i - \Delta, Q_i + \Delta]$ with a unit cost of α.

Finally, for every pair of bins (i, j), add an edge (p_i, q_j) with lower bound $b = 0$, capacity $c = \infty$, and cost $w = \mathbf{M}_{i,j}$. For example, suppose $\mathbf{M}_{i,i} = 0 \quad \forall i$ and only $\mathbf{M}_{1,2}$ is defined, then the resulting graph as Figure 7.

Now we run the min-cost flow with lower bounds solver on this graph to get the AEMC. Moreover, since the number of cells in P and Q are the same, and the cost of not moving bins is 0 (i.e.

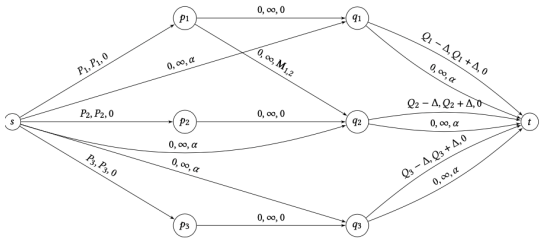


Figure 7: Adding lower bounds, capacities and costs.

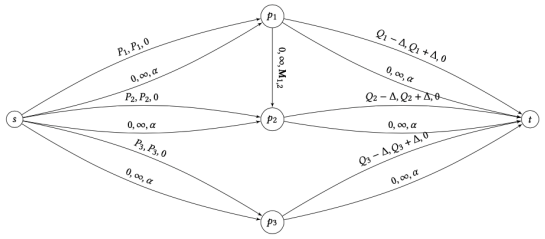


Figure 8: Compressing the p_i and q_i nodes

$\mathbf{M}_{i,i} = 0 \quad \forall i$), we can compress the p_i and q_i nodes together to reduce the number of nodes by half, as shown in Figure 8.

A.2 Solving Min-cost Flow with Lower Bounds

We solve the min-cost flow with lower bounds problem by reducing it to the *min-cost circulation* problem. The difference is that there are no source and sink in the graph in this new problem. Any min-cost flow with lower bounds problem can be reduced to an instance of the min-cost circulation problem by adding an edge (t, s) with lower bound $b(t, s) = 0$, capacity $c(t, s) = \infty$, and unit cost $w(t, s) = 0$.

Let us define another problem called *min-cost circulation with node demands*: You are given a graph $G = (V, E)$, where each edge (u, v) has a capacity $c(u, v)$ and a unit cost $w(u, v)$, and each node has a demand $d(u)$. We need to assign a flow to each edge (u, v) such that each node's demand is satisfied, and the overall circulation is minimal. Formally, the problem can be described as the following linear program:

$$\begin{aligned} \min \quad & \sum_{(u,v)\in E} f(u,v) \cdot w(u,v) \\ \text{s.t. } & 0 \le f(u,v) \le c(u,v) && \forall (u,v) \in E \\ & \sum_{(x,u)\in E} f(x,u) - \sum_{(u,y)\in E} f(u,y) = d(u) && \forall u \in V \end{aligned}$$

We can transform an instance of min-cost circulation with lower bounds to an instance of min-cost circulation with node demands using the following reduction:

- First, initialize the demands for all nodes to be zero, i.e. $d(u) = 0$.
- For each edge (u, v), add $b(u, v) \cdot w(u, v)$ to the answer, add $b(u, v)$ to $d(u)$, subtract $b(u, v)$ from $d(v)$, and reform this

edge to have no lower bound and with a capacity of $c(u, v) - b(u, v)$.

Intuitively, since each edge (u, v) must have a lower bound of $b(u, v)$, we simply force this lower bound (thus we need to add $b(u, v) \cdot w(u, v)$ to the answer); this implies that the remaining flexible capacity for this edge is only $c(u, v) - b(u, v)$. To enforce that there must be such flow going through that edge, we reserve a supply of $b(u, v)$ on the node u, therefore u must "absorb" $b(u, v)$ flow from other edges, hence adding $b(u, v)$ to $d(u)$; similarly, v must "release" $b(u, v)$ to other edges, hence subtracting $b(u, v)$ from $d(v)$.

The min-cost circulation with node demands problem is a well known problem, with the state of the art algorithm having a time complexity of $O(mn \log U \log(nC))$, where $n = |V|$, $m = |E|$, $U = \max w(u, v)$, and $C = \max c(u, v)$[3]. Our implementation uses OR-Tools solver, which has a time complexity of $O(n^2 m \log(nC))$.

B DETAILED RESULTS OF EXPERIMENTS

Table 1 shows different scores of synthetic datasets. The "mediocre" and "very-poor" datasets are from Example Temporal Map Data provided by the competition. The synthetic datasets, "PrivSyn-X", are generated by algorithm in [33] with privacy budget $\epsilon = X$ and truncation which limits the per-user contribution to 2. The "lap-X" datasets are produced by basic Laplace mechanism in differential privacy with privacy budget $\epsilon = X$ and sensitivity is set to 20. The "sample-X" is for randomly sampling X portion of incidents from the incident datasets.

The Inv Pie score is computed by $(1 - S/3336)$, where S is the pie-chart score of the dataset. The range query (RQ) is the relative error of 300 randomly generated range queries, each of which queries on the count in 30% of month, neighborhood and incident types.

Dataset	Inv Pie	RQ	AEMC						
			M	N	I	M+I	M+N	N+I	3-way
mediocre	0.5689	1.0666	0.8669	0.8666	0.8667	0.8641	0.8624	0.8117	0.2667
very-poor	0.96	9.0631	7.6956	7.6952	7.6954	7.6927	7.691	7.6401	7.1258
PrivSyn-0.1	0.8236	0.0451	0.026	0.0598	0.1026	0.0748	0.0755	0.1169	0.1013
PrivSyn-0.25	0.6751	0.0175	0.0086	0.0214	0.0413	0.0286	0.0266	0.0616	0.0556
PrivSyn-0.5	0.5801	0.0103	0.0048	0.009	0.018	0.0131	0.0125	0.0332	0.0361
PrivSyn-1.0	0.5222	0.007	0.0022	0.0044	0.0071	0.0053	0.0056	0.0157	0.0305
PrivSyn-2.0	0.5039	0.0074	0.0011	0.002	0.0029	0.0025	0.0026	0.0061	0.0298
PrivSyn-4.0	0.5113	0.0169	0.0142	0.0138	0.014	0.0117	0.0097	0.0016	0.0296
PrivSyn-10.0	0.5017	0.007	0.0019	0.0028	0.0034	0.0025	0.0025	0.0053	0.03
lap-0.25	0.9853	19.3996	15.6775	15.6772	15.6773	15.6747	15.673	15.6216	15.1237
lap-0.5	0.9619	8.8361	7.6918	7.6915	7.6916	7.689	7.6873	7.6366	7.1222
lap-1.0	0.9072	4.3788	3.7608	3.7604	3.7606	3.7579	3.7562	3.706	3.1763
lap-2.0	0.8278	2.115	1.8199	1.8195	1.8196	1.817	1.8153	1.7651	1.224
lap-4.0	0.5701	1.0375	0.8661	0.8658	0.8659	0.8634	0.8616	0.8115	0.2661
lap-10.0	0.3935	0.369	0.3178	0.3174	0.3176	0.3151	0.3132	0.2622	0.0481
sample-0.01	0.8883	0.0407	0.019	0.046	0.0252	0.0461	0.0827	0.1428	0.1956
sample-0.05	0.821	0.0152	0.0083	0.0199	0.0096	0.0197	0.0344	0.0669	0.111
sample-0.1	0.7337	0.0122	0.0059	0.0145	0.0075	0.0142	0.0244	0.0443	0.074
sample-0.25	0.5494	0.0072	0.0028	0.008	0.0044	0.0077	0.0135	0.0222	0.0315
sample-0.5	0.3834	0.0042	0.0019	0.0045	0.0022	0.0043	0.0073	0.0103	0.0106

Table 1: Compare inversed Pie-chart metric (Inv Pie), range query (RQ) and AEMC on different marginals, M: "months", N: "neighborhood", I: "incident type". $\Delta = 2$ **for AEMC.**