Preserving Diversity in Anonymized Data

Mostafa Milani
The University of Western Ontario
London, Ontario, Canada
mostafa.milani@uwo.ca

Yu Huang
McMaster University
Hamilton, Ontario, Canada
huang223@mcmaster.ca

Fei Chiang
McMaster University
Hamilton, Ontario, Canada
fchiang@mcmaster.ca

ABSTRACT

Recent privacy legislation has aimed to restrict and control the amount of personal data published by companies and shared with third parties. Much of this real data is not only sensitive requiring anonymization but also contains characteristic details from a variety of individuals. This diversity is desirable in many applications ranging from Web search to drug and product development. Unfortunately, data anonymization techniques have largely ignored diversity in its published result. This inadvertently propagates underlying bias in subsequent data analysis. We study the problem of finding a diverse anonymized data instance where diversity is measured via a set of diversity constraints. We formalize diversity constraints, and present a clustering-based algorithm for finding a diverse anonymized instance. We show the effectiveness and efficiency of our techniques against existing baselines. Our work aligns with recent trends towards responsible data science by coupling diversity with privacy-preserving data publishing.

1 INTRODUCTION

Organizations often share user information with third parties to analyze collective user behaviour and for targeted marketing. Protecting user privacy is critical to safeguard personal and sensitive data. The European Union General Data Protection Regulation (GDPR), and variants such as the California Consumer Protection Act (CCPA) aim to control how organizations manage user data. For example, a major tenet in GDPR is *data minimization* that states companies should collect and share only a minimal amount of personal data sufficient for their purpose. Given the impossibility of knowing how a published data instance will be used in the future, determining a minimal amount of personal data to share is a challenge.

Privacy-preserving data publishing (PPDP) safeguards individual privacy while ensuring the published data remains practically useful for analysis. Anonymization is the most common form of PPDP, where quasi-identifiers and/or sensitive values are obfuscated via suppression or generalization [11]. As anonymized instances are shared with third parties for decision making and analysis, there is growing interest to ensure that data (and the algorithms that generate and use the data) are diverse and fair. Diversity is a rather established notion in data analytics that refers to the property of a selected set of individuals. Diversity requires the selected set to have a minimum representation from each group of individuals [9, 23] while determining the minimum bound for each group is often domain and user dependent.

To avoid biased decision making, incorporating diversity into computational models is essential to prevent and minimize discrimination against minority groups. In this paper, we focus on diversity, and study how diversity requirements can be modeled and satisfied in PPDP. In PPDP, non-diverse data instances that obfuscate characteristic attributes of a minority group give an inaccurate representation of the population in subsequent data analysis. Unfortunately, early PPDP work [11, 22, 24], and recent work on PPDP for graphs [12, 13], and interactive settings [14, 15] have not considered diversity in published instances.

Example 1.1. Table 1 shows relation R containing patients' medical records describing gender (GEN), ethnicity (ETH), age (AGE), province (PRV), city (CTY), and diagnosed disease (DIAG). Third-parties such as pharmaceuticals, insurance firms are interested in an anonymized R containing patients from diverse geographies, gender, and ethnicities. Let GEN, ETH, AGE, PRV, CTY be quasi-identifier (QI) attributes, and let DIAG be a sensitive attribute. Existing PPDP methods such as k-anonymity prevent re-identification of an individual along the QI attributes from $k-1$ other tuples. Table 2 shows a k-anonymized instance for $k=3$ where tuples are clustered along the QI attributes via value suppression [22, 24].

The k-anonymization problem is to generate a k-anonymous relation through an anonymization process, such as generalization and suppression, while incurring minimum information loss. Suppression replaces some QI attribute values with $\star$s to achieve k-anonymity, and is often considered to be a maximal form of generalization that obscures a value completely. There are several measures of information loss [7, 11], e.g., counting the number of $\star$s. Existing k-anonymization techniques do not preserve diversity in R since these measures do not capture diversity. □

Unfortunately, existing methods fail to provide any diversity guarantees in published, privatized data instances, leading to inaccurate and biased decision making. For example, in Table 2, we have lost the African and Caucasian ethnicity from the (second) group of Male, and the Female gender from the (first) group of Caucasian. These records, which exclude characteristic features of minority groups, misrepresent the true patient population. Efforts to obtain a diverse instance of patients, for example, to obtain minimum proportions of patients along GEN and ETH attributes, are hindered due to these missing values.

To model diversity, existing work has proposed declarative methods in the form of *diversity constraints*, which define the expected frequencies that characteristic values (of a group) must satisfy [23]. Similar to previous work, we consider one or more discrete value attributes to be of particular concern, and we define diversity with respect to the values of these attributes. Using k-anonymity as our privacy definition, and given a relation R, constant k, and a set of diversity constraints Σ, we study the problem of publishing a k-anonymized and diverse instance R^*. An example of a diversity constraint $\sigma_1 = (ETH[Asian], 2, 5)$ requires an anonymized instance to contain between two and five Asian individuals, which is satisfied by Table 1 and Table 2. Diversity constraints provide a declarative definition of the minimum and maximum frequency bounds that specific attribute domain values should appear in R^* [23].

ID	GEN	ETH	AGE	PRV	CTY	DIAG
t_1	Female	Caucasian	80	AB	Calgary	Hypertension
t_2	Female	Caucasian	32	AB	Calgary	Tuberculosis
t_3	Male	Caucasian	59	AB	Calgary	Osteoarthritis
t_4	Male	Caucasian	46	MB	Winnipeg	Migraine
t_5	Male	African	32	MB	Winnipeg	Hypertension
t_6	Male	African	43	BC	Vancouver	Seizure
t_7	Male	Caucasian	35	BC	Vancouver	Hypertension
t_8	Female	Asian	58	BC	Vancouver	Seizure
t_9	Female	Asian	63	MB	Winnipeg	Influenza
t_{10}	Female	Asian	71	BC	Vancouver	Migraine

Table 1: Medical records relation (R)

ID	GEN	ETH	AGE	PRV	CTY	DIAG
r_1	⋆	Caucasian	⋆	AB	Calgary	Hypertension
r_2	⋆	Caucasian	⋆	AB	Calgary	Tuberculosis
r_3	⋆	Caucasian	⋆	AB	Calgary	Osteoarthritis
r_4	Male	⋆	⋆	⋆	⋆	Migraine
r_5	Male	⋆	⋆	⋆	⋆	Hypertension
r_6	Male	⋆	⋆	⋆	⋆	Seizure
r_7	Male	⋆	⋆	⋆	⋆	Hypertension
r_8	Female	Asian	⋆	⋆	⋆	Seizure
r_9	Female	Asian	⋆	⋆	⋆	Influenza
r_{10}	Female	Asian	⋆	⋆	⋆	Migraine

Table 2: Anonymized relation with $k = 3$

ID	GEN	ETH	AGE	PRV	CTY	DIAG
g_1	Female	Caucasian	⋆	AB	Calgary	Hypertension
g_2	Female	Caucasian	⋆	AB	Calgary	Tuberculosis
g_3	Male	Caucasian	⋆	⋆	⋆	Osteoarthritis
g_4	Male	Caucasian	⋆	⋆	⋆	Migraine
g_5	Male	African	⋆	⋆	⋆	Hypertension
g_6	Male	African	⋆	⋆	⋆	Seizure
g_7	⋆	⋆	⋆	BC	Vancouver	Hypertension
g_8	⋆	⋆	⋆	BC	Vancouver	Seizure
g_9	Female	Asian	⋆	⋆	⋆	Influenza
g_{10}	Female	Asian	⋆	⋆	⋆	Migraine

Table 3: Anonymized relation with $k = 2$.

We define the (k, Σ)-anonymization problem, which seeks an optimal k-anonymous instance R^* that satisfies a set of diversity constraints Σ. We propose the *DIVA* algorithm to compute a DIVerse and Anonymized R^*. *DIVA* integrates anonymization with diversity by applying value suppression to find a k-anonymous instance satisfying a set of diversity constraints.

Contributions. We make the following contributions:

(1) We formalize diversity constraitns in PPDP, and we define the (k, Σ)-anonymization problem that seeks a k-anonymous relation with value generalization that satisfies Σ.
(2) We introduce *DIVA*, a clustering-based algorithm that solves the (k, Σ)-anonymization problem with minimal suppression.
(3) We evaluate the effectiveness and efficiency of our selection strategies over the basic version of *DIVA*. We show that *DIVA* achieves improved performance over existing baselines.

2 PRELIMINARIES

Basic Notations. A *relation* R with a *schema* $\mathcal{R} = \{A_1, ..., A_n\}$ is a finite set of n-ary tuples $\{t_1, ..., t_N\}$. A, B, C refer to single attributes and X, Y, Z as sets of attributes.

Privacy-Preserving Data Publishing. k-anonymity prevents re-identification of an individual in an anonymized data set [22, 24]. Attributes in a relation are either *identifiers* such as SSN that uniquely identify an individual, *quasi-identifier* (QI) attributes such as ethnicity, address, age that together can identify an individual, or *sensitive* attributes that contain personal information.

Definition 2.1 (QI-group and k-anonymity). A relation R is k-anonymous if every QI-group has at least k tuples. A QI-group is a set of tuples with the same values in the QI attributes. □

For example, Table 2 has three QI-groups, $\{r_1, r_2, r_3\}$, $\{r_4, r_5, r_6, r_7\}$, and $\{r_8, r_9, r_{10}\}$, and is 3-anonymous. Extensions of k-anonymity include l-diversity, t-closeness, and (X, Y)-anonymity, which provide improved privacy confidence (cf. [11] for a survey). We apply k-anonymity for its ease of presentation, however, our definitions and techniques are extensible to include recent PPDP models.

Suppression. Suppression generates an anonymized relation R' from a relation R by replacing some QI values in R with ⋆. We denote this by $R \sqsubseteq R'$. Suppression clearly causes information loss which is typically measured by the number of ⋆s in R'.

Definition 2.2 (k-anonymization problem [24]). Given R, the k-anonymization problem is to find R^* such that (1) $R \sqsubseteq R^*$; (2) R^* is k-anonymous; and (3) R^* has minimum information loss. □

Diversity Constraints. Diversity constraints are originally proposed for the set selection problem defined as follows [23]. Given a set of N items, each associated with a characteristic attribute and a utility score, the *set selection problem* is to select M items to maximize a utility score subject to diversity constraints. The utility score is the sum of scores of each selected item. Let there be d distinct values of the characteristic attribute and m_i with $i \in [1, d]$ be the number of selected items with each distinct value such that $m_i \in [0, M]$ and $\sum_i(m_i) = M$. A diversity constraint ϕ of the form $floor_i \leq m_i \leq ceiling_i$ specifies upper and lower bounds on m_i, i.e. the number of items with the i-th characteristic value. These constraints ensure representation from each category known as coverage-based diversity. To avoid tokenism, where there is only a single representative from each category, we can increase the lower bound, e.g., $m_i > 1$.

Problem Definition. We apply the concept of diversity constraints as proposed by Stoyanovich et. al [23], and introduce a formal definition of diversity constraints in relational data.

Definition 2.3 (Diversity Constraints). A diversity constraint over a relation schema $\mathcal{R}$ is of the form $\sigma = (A[a], \lambda_l, \lambda_r)$ in which $A \in \mathcal{R}$, $a \in dom(A)$ and λ_l, λ_r are non-negative integers. The diversity constraint σ is satisfied by a relation R of schema $\mathcal{R}$ denoted $R \models \sigma$ if and only if there are at least λ_l and at most λ_r occurrences of the value a in attribute A of relation R. We call $[\lambda_l, \lambda_r]$ the frequency range and $A[a]$ the characteristic (or target) value of σ. A set of diversity constraints Σ is satisfied by R, denoted by $R \models \Sigma$, iff R satisfies every $\sigma \in \Sigma$. □

Diversity constraints can be extended to multiple attributes by replacing $A[a]$ with $X[t]$, where X is a set of attributes and t is a tuple with values from these attributes. This extended diversity constraint $\sigma = (X[t], \lambda_l, \lambda_r)$ is satisfied by R if there are at least λ_l and at most λ_r tuples in R with the same attribute values in t. To validate that R satisfies σ, we can run a query that counts the number of occurrences of the target values t in attributes X of R and then check if this number lies in the frequency range $[\lambda_l, \lambda_r]$. Given a set of diversity constraints Σ, we define our problem.

Definition 2.4 (Problem Statement ((k, Σ)-anonymization)). Consider a relation R, a constant k, a set of diversity constraints Σ. The (k, Σ)-*anonymization problem* is to find a relation R^* where: (1) $R \sqsubseteq R^*$, (2) R^* is k-anonymous, (3) $R^* \models \Sigma$, and (4) R^* has minimal information loss, i.e., a minimum number of ⋆'s. □

3 THE DIVA ALGORITHM

We present the DIVersity and Anonymization algorithm (*DIVA*) that solves the (k, Σ)-anonymization problem. *DIVA* takes as input a relation R, a set of diversity constraints Σ, constant k, and returns a k-anonymous and diverse relation R' that satisfies Σ. *DIVA* work in two phases: (i) *clustering*, by partitioning R into disjoint clusters of size greater than or equal to k, while considering Σ; and (ii) *suppression*, by suppressing a minimal number of QI values in each cluster such that they have the same QI values, and form a QI-group of size $\geq k$. The result is a k-anonymous relation, as every QI-group is of size $\geq k$.

3.1 Overview

Figure 1 presents an overview of *DIVA*. The algorithm begins in *DiverseClustering*, which generates a diverse clustering $\mathcal{S}_\Sigma$ that

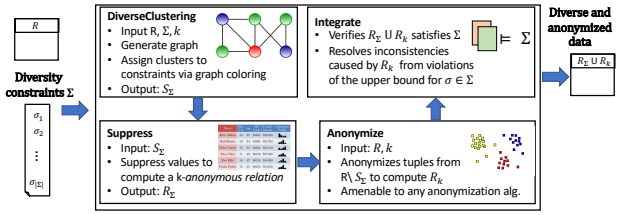


Figure 1: *DIVA* overview.

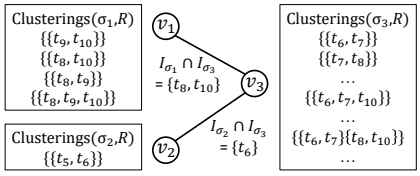


Figure 2: Graph representation of constraints.

clusters tuples in R such that each $\sigma \in \Sigma$ is satisfied. We note that the clustering $\mathcal{S}_\Sigma$ may involve a subset of tuples in R, which are necessary to satisfy Σ. In *Suppress*, *DIVA* anonymizes the tuples in $\mathcal{S}_\Sigma$ using value suppression. The result is a relation R_Σ that satisfies Σ and is k-anonymous, however, may not represent all tuples in R. In the *Anonymize* step, *DIVA* runs an existing off-the-shelf anonymization algorithm to generate R_k by anonymizing the tuples of R that do not exist in $\mathcal{S}_\Sigma$ and R_Σ. Lastly, the *Integrate* phase integrates R_Σ and R_k to validate that $R_\Sigma \cup R_k$ doesn't violate the constraints' upper bounds.

Algorithm 1 presents *DIVA* details. *DiverseClustering* is a search algorithm that generates the diverse clustering $\mathcal{S}_\Sigma$. For each diversity constraint $\sigma \in \Sigma$, it computes a clustering that ensures the satisfaction of σ. If no diverse clustering exists, i.e., there is no diverse k-anonymous relation R', *DiverseClustering* returns $\mathcal{S}_\Sigma := \emptyset$, and *DIVA* returns an error. Algorithm 2 provides *Suppress* details that takes a clustering $\mathcal{S}$ and returns a relation R_s. For every tuple t in a cluster $C \in \mathcal{S}$, there is a corresponding tuple $r \in R_s$ with the same sensitive values as t. For every QI attribute A_i, $r[A_i]$ is suppressed, i.e. $r[A_i] = \star$, if the tuples in C have different values of A_i (Line 4). Due to this value suppression, R_s contains QI groups corresponding to the clusters in $\mathcal{S}$. In Line 3, we call *Suppress* with input $\mathcal{S}_\Sigma$ and output R_Σ.

Returning to Algorithm 1, there may be tuples of R that do not exist in $\mathcal{S}_\Sigma$. The *Anonymize* routine anonymizes these remaining tuples by applying an existing k-anonymization algorithm to compute R_k. In our evaluation, we use the k-member algorithm [6], but *DIVA* is amenable to any k-anonymization algorithm. Lastly, *Integrate* computes $R' = R_\Sigma \cup R_k$ and checks whether $R' \models \Sigma$. If there exists a violation, R' falsifies the upper bound of some constraint(s) in Σ due to R_k. *Integrate* resolves this by suppressing minimal values in R' to satisfy Σ. We return R' as the final output.

Example 3.1. Consider an execution of *DIVA* with relation R in Table 1, $k = 2$, and $\Sigma = \{\sigma_1, \sigma_2, \sigma_3\}$, where $\sigma_1 = (ETH[Asian], 2, 5)$, $\sigma_2 = (ETH[African], 1, 3)$, $\sigma_3 = (CTY[Vancouver], 2, 4)$. *DiverseClustering* returns $\mathcal{S}_\Sigma = \{C_1, C_2, C_3\}$ where $C_1 = \{t_9, t_{10}\}$, $C_2 = \{t_5, t_6\}$, and $C_3 = \{t_7, t_8\}$. R_Σ contains suppressed tuples $g_5, ..., g_{10}$ in Table 3 with QI groups $\{g_5, g_6\}$, $\{g_7, g_8\}$, and $\{g_9, g_{10}\}$ that correspond to the clusters in $\mathcal{S}_\Sigma$. R_Σ satisfies Σ where each QI group satisfies a constraint. The *Anonymize* procedure generates $R_k = \{g_1, ..., g_4\}$, and the *Integrate* procedure returns $R' = R_k \cup R_\Sigma$ (Table 3) which is $k = 2$-anonymous and satisfies Σ. □

Algorithm 1: *DIVA* (R, Σ, k)

Output: k-anonymous and diverse relation.
1 $\mathcal{S}_\Sigma := DiverseClustering(R, \Sigma, k)$;
2 **if** $\mathcal{S}_\Sigma = \emptyset$ **then return "relation does not exist"**;
3 $R_\Sigma := Suppress(\mathcal{S}_\Sigma)$;
4 **foreach** $C_i \in \mathcal{S}_\Sigma$ **do** $R := R \setminus C_i$;
5 $R_k := Anonymize(R, k)$;
6 **return** $Integrate(R_\Sigma, R_k)$;

Algorithm 2: *Suppress*$(\mathcal{S})$

Input: A clustering $\mathcal{S}$ of tuples with schema $\mathcal{R}$.
Output: Relation R_s (initialized to $\emptyset$.
1 **foreach** $C \in \mathcal{S}$ **and** $t \in C$ **do**
2 $r := t$; $R_s := R_s \cup \{r\}$;
3 **foreach QI Attribute** $A_i \in \mathcal{R}$ **do**
4 **if** $|C[A_i]| > 1$ **then** $r[A_i] := \star$;
5 **return** R_s;

3.2 Diverse Clustering

We describe how *DiverseClustering* computes a clustering $\mathcal{S}_\Sigma$ that satisfies Σ. We first define the semantics of how a clustering satisfies a diversity constraint.

Definition 3.2. Given σ defined over R, and a clustering $\mathcal{S}$, $\mathcal{S}$ satisfies σ, denoted as $\mathcal{S} \Vdash \sigma$, if $Suppress(\mathcal{S}) \models \sigma$. The clustering $\mathcal{S}$ satisfies a set of constraints Σ, if $\mathcal{S} \Vdash \sigma_i$ for every $\sigma_i \in \Sigma$. □

Intuitively, $\mathcal{S} \Vdash \sigma$ if the relation returned from $Suppress(\mathcal{S})$ in Algorithm 2 satisfies σ according to Definition 2.3. In Example 3.1, $\mathcal{S}_\Sigma = \{\{t_5, t_6\}, \{t_7, t_8\}, \{t_9, t_{10}\}\} \Vdash \Sigma$ because $Suppress(\mathcal{S}_\Sigma) = \{g_5, ..., g_{10}\} \models \Sigma$.

DiverseClustering finds $\mathcal{S}_\Sigma \Vdash \Sigma$ by computing clusterings $\mathcal{S}_{\sigma_i}$ that satisfy each diversity constraint $\sigma_i \in \Sigma$ and merging them to generate $\mathcal{S}_\Sigma$. In Example 3.1, $\mathcal{S}_{\sigma_1} = \{C_1\} = \{t_9, t_{10}\} \Vdash \sigma_1$, $\mathcal{S}_{\sigma_2} = \{C_2\} = \{t_5, t_6\} \Vdash \sigma_2$, $\mathcal{S}_{\sigma_3} = \{C_3\} = \{t_7, t_8\} \Vdash \sigma_3$, and $\mathcal{S}_\Sigma = \mathcal{S}_{\sigma_1} \cup \mathcal{S}_{\sigma_2} \cup \mathcal{S}_{\sigma_3} = \{C_1, C_2, C_3\} \Vdash \Sigma$. Two conditions must hold while we search for $\mathcal{S}_{\sigma_i}$. First, the clusters in $\mathcal{S}_{\sigma_i}$ must be disjoint, unless they are equal. Specfically, for every pair of clusters $C \in \mathcal{S}_{\sigma_i}$ and $C' \in \mathcal{S}_{\sigma_j}$, either $C' \cap C = \emptyset$ or $C' = C$. For overlapping cluster pairs, the result of *Suppress* will not form QI groups. If we merge the clusters, then the result may not satisfy the conditions. For example, if $\mathcal{S}_{\sigma_2} = \{\{t_5, t_6\}\}$, $\mathcal{S}_{\sigma_3} = \{\{t_6, t_7\}\}$, and $\Sigma = \{\sigma_2, \sigma_3\}$, and we merge them into $\mathcal{S}_\Sigma = \{\{t_5, t_6, t_7\}\}$, then $\mathcal{S}_\Sigma \nVdash \Sigma$, although $\mathcal{S}_{\sigma_2} \Vdash \sigma_2$ and $\mathcal{S}_{\sigma_3} \Vdash \sigma_3$. Second, we select each $\mathcal{S}_{\sigma_i}$ such that it does not falsify the upper bounds of other constraints. In Example 3.1, consider $\Sigma = \{\sigma_2, \sigma_4\}$ with a new constraint $\sigma_4 = (GEN[\text{Male}], 1, 3)$, which requires at least one but not more than 3 men. Then, $\{\{t_5, t_6\}\} \Vdash \sigma_2$, and $\{\{t_3, t_4\}\} \Vdash \sigma_4$, but $\{\{t_5, t_6\}, \{t_3, t_4\}\} \nVdash \{\sigma_2, \sigma_4\}$ since the upper bound of σ_4 is falsified. The clustering $\mathcal{S}_{\sigma_2}$ preserves two more Male values, and falsifying the upper bound in σ_4. This means we cannot build the $\mathcal{S}_{\sigma_i}$ separately, and we must consider the interactions between the constraints. This is clearly a local condition where choosing each $\mathcal{S}_{\sigma_i}$, we consider the related constraints that have overlapping tuples with σ_i, and use this property to convert diverse clustering to graph coloring.

3.3 Modeling as Graph Coloring

Given an undirected graph $G = (\Gamma, E)$, where Γ and E denote the set of nodes and edges, respectively, and m distinct colors, the graph coloring problem is to color all nodes subject to certain

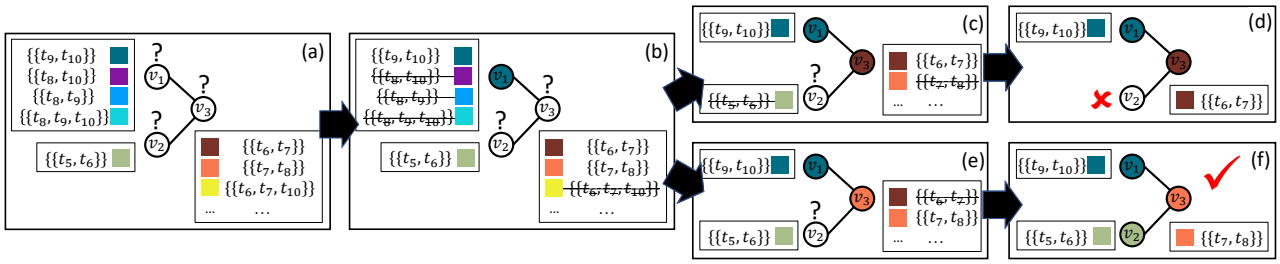


Figure 3: Graph coloring for diverse clustering. Nodes are colored in the order v_1, v_3, v_2.

Algorithm 3: *DiverseClustering*(R, Σ, k)

Output: Clustering $\mathcal{S}_\Sigma$.

1 $G :=$ *BuildGraph*(R, Σ); $V := \emptyset$; $\mathcal{S}_\Sigma := \emptyset$;
2 **if** *Coloring*(G, V, R) **then**
3 | **foreach** $\langle v_i, c_i \rangle \in V$ **do** $\mathcal{S}_\Sigma := \mathcal{S}_\Sigma \cup c_i.clustering$;
4 **return** $\mathcal{S}_\Sigma$;

Algorithm 4: *Coloring*(G, V, R)

Output: true if there is a coloring of G, otherwise **false**.

1 **if** V **contains all nodes of** G **then return true**;
2 $v :=$ *NextNode*(G, V);
3 **foreach** $\mathcal{S} \in$ *Clusterings*$(v.constraint, R)$ **do**
4 | **if** *IsConsistent*$(\mathcal{S}, v)$ **then**
5 | | $c :=$ **new color with clustering** $\mathcal{S}$;
6 | | $V := V \cup \{\langle v, c \rangle\}$;
7 | | **if** *Coloring*(G, V, R) **then return true** ;
8 | | $V := V \setminus \{\langle v, c \rangle\}$;
9 **return false**

constraints, e.g., no two adjacent nodes can have the same color. For relation R and diversity constraints Σ, we model each diversity constraint $\sigma_i \in \Sigma$ as a node $v_i \in \Gamma$. We use $v_i.constraint$ to refer to σ_i. An undirected edge $e_{ij} = \{v_i, v_j\} \in E$ exists between nodes v_i and v_j if σ_i and σ_j have overlapping target tuples, i.e., $I_{\sigma_i} \cap I_{\sigma_j}$, where the target tuples of a constraint σ_i, denoted by I_{σ_i}, is the set of tuples in R that have the target values in σ_i.

Example 3.3. Figure 2 shows G with three nodes corresponding to $\sigma_1, \sigma_2, \sigma_3$, and each of their neighboring constraints modeled via edges $E = \{\{v_1, v_3\}, \{v_2, v_3\}\}$. The edge labels show the non-empty intersections between their target tuple sets. The sets of target tuples are $I_{\sigma_1} = \{t_8, t_9, t_{10}\}$, $I_{\sigma_2} = \{t_5, t_6\}$, and $I_{\sigma_3} = \{t_6, t_7, t_8, t_{10}\}$, which means σ_1, σ_3 and σ_2, σ_3 are neighboring constraints, but there is no edge between v_1 and v_2 because $I_{\sigma_1} \cap I_{\sigma_3} = \{t_8, t_{10}\}$ and $I_{\sigma_2} \cap I_{\sigma_3} = \{t_6\}$, and $I_{\sigma_1} \cap I_{\sigma_2} = \emptyset$. Beside each node, we show the clusterings that satisfy the corresponding constraint.

Choosing a color for node v_i is analogous to finding a clustering $\mathcal{S}_{\sigma_i}$ for σ_i. The goal is to color all nodes, while the color of each node is *consistent* with the color of its neighboring nodes. This means the corresponding clusterings must satisfy the two conditions that we mentioned earlier. For a color c, $c.clustering$ refers to its corresponding clustering.

Algorithm 3 presents the details of *DiverseClustering*. We build the graph G for Σ and R (Line 1). We then initialize the clustering $\mathcal{S}_\Sigma$ and a mapping V that stores the color (assigned clustering) for each node, and check if a coloring exists via *Coloring*.

Algorithm 4 presents the recursive function, *Coloring*, that takes a graph G, the mapping V (specifying the colored nodes), relation R, and returns *true* if the remaining nodes of G can be colored; otherwise it returns *false*. Note that choosing a color for a node, can restrict the choice of colors for neighboring nodes when the clusterings have overlap. *Coloring* iterates over every uncolored node and assigns a color (clustering) that is consistent with its neighboring nodes (constraints). Specifically, we check that the two search conditions mentioned earlier are satisfied. We propose three versions of *DIVA*: (1) DIVA-Basic: *Coloring* randomly selects an uncolored node (Line 2) to color using *NextNode*; (2) MaxFanOut: selects constraints with a minimal number of clusterings; and (3) MinChoice: selects constraints with a maximal overlap with neighboring constraints. We describe the latter two versions later in this section.

Given a node v, we color v by checking whether its candidate clustering, and its adjacent nodes are consistent (Alg. 4, Lines 3-8). The *Clusterings* routine returns minimal clusterings $\mathcal{S}$ that satisfy $v.constraint$ ($Suppress(\mathcal{S}) \vDash v.constraint$). For example, $Clusterings(\sigma_1, R)$ contains four clusterings $\{\{t_8, t_9\}\}$, $\{\{t_8, t_{10}\}\}$, $\{\{t_9, t_{10}\}\}$, $\{\{t_8,\ t_9, t_{10}\}\}$, while $Clusterings(\sigma_2, R)$ contains one clustering $\{\{t_5, t_6\}\}$. In Lines 4-8, we check whether $\mathcal{S}$ is consistent with the clusterings of the neighboring constraints. If so, we assign a new color c to the clustering $\mathcal{S}$, and we temporarily color v with c by adding $\langle v, c \rangle$ to V. We then recursively call *Coloring* to check whether the remaining nodes in G can be colored. If color c does not work, i.e. *Coloring* returns false, we remove $\langle v, c \rangle$ from V, and try another color. If all clusterings are inconsistent, i.e., there is no successful coloring of v, we return false (Line 9), to backtrack and evaluate a different node. To compute a satisfying clustering depends not only on k, and the frequency of characteristic values in R with respect to $[\lambda_l, \lambda_r]$ in σ, but also on the distribution of these characteristic values. We empirically study the impact of data distribution on accuracy in Section 4.

Example 3.4. Figure 3 shows an execution of *Coloring* over graph G with nodes $\{v_1, v_3, v_2\}$. Figure 3(a) initializes nodes as uncolored. We first consider v_1, and select $S_{\sigma_1} = \{\{t_9, t_{10}\}\}$ (Figure 3(b)). We color nodes v_2 and v_3 by recursively calling *Coloring*. Coloring one node may restrict the color choice of neighboring nodes, e.g. after we select $\{\{t_9, t_{10}\}\}$ for v_1, we cannot select $\{\{t_6, t_7, t_{10}\}\}$ for v_3 due to the overlapping tuple t_{10}. For node v_3, we have several choices including $\{\{t_6, t_7\}\}$ and $\{\{t_7, t_8\}\}$. In Figure 3(c), we assume the coloring algorithm chooses $\{\{t_6, t_7\}\}$ for v_3. As a result, $\{\{t_5, t_6\}\}$, which was the only choice for v_2, cannot be used due to the overlapping tuple t_6. This leads the algorithm towards an unsatisfying clustering (Figure 3(d)). *DIVA* backtracks its last decision for v_3 by selecting a different color, $\{\{t_7, t_8\}\}$ for v_3 in Figure 3(e). In this case, the clustering $\{\{t_5, t_6\}\}$ for v_2 does not overlap with $\{\{t_7, t_8\}\}$. Since we have found a clustering that satisfies all the constraints (i.e., a coloring of all nodes), *Coloring* returns *true* with V containing the nodes and their colors (i.e., clusterings). *DiverseClustering* uses V to compute the final clustering as $\mathcal{S}_\Sigma = \{\{t_5, t_6\}, \{t_7, t_8\}, \{t_9, t_{10}\}\}$. □

Table 4: Data characteristics.

	Pantheon	Census	Credit	Pop-Syn
$\|R\|$	11,341	299,285	1000	100,000
n	17	40	20	7
$\|\Pi_{QI}(R)\|$	5,636	12,405	60	24,630
$\|\Sigma\|$	24	21	18	10

Table 5: Parameter values (defaults in bold)

Symbol	Description	Values
$\|R\|$	#tuples	60k, 120k, **180k**, 240k, 300k
$\|\Sigma\|$	#constraints	4, **8**, 12, 16, 20
$cf(\Sigma)$	conflict rate	0, **0.2**, 0.4, 0.6, 0.8, 1
k	minimum cluster size	10, **20**, 30, 40, 50

DIVA runs in polynomial time w.r.t. the size of R. The runtime is split between *DiverseClustering* and *Anonymize*. *DiverseClustering* runs in polynomial time as the number of clusters considered in coloring for each constraint is polynomial w.r.t. R. *Anonymize* runs an off-the-shelf algorithm. We use the k-member anonymization algorithm, which runs in polynomial time [6].

Selection Strategies. In the basic version of *DIVA*, we randomly select a constraint and a clustering to evaluate. These choices impact performance as poor initial selections can lead to increased backtracking operations downstream. We selectively order the constraints (nodes) and clusterings (colors) that most likely lead to a graph coloring while minimizing the need to backtrack.

MinChoice: We select the most restrictive constraints first, i.e., in *Nextnode*, we select v with minimum $|Clusterings(v.constraint, R)|$. As we visit nodes and assign (colors) clusterings, we update the candidate clusterings for their neighbors.

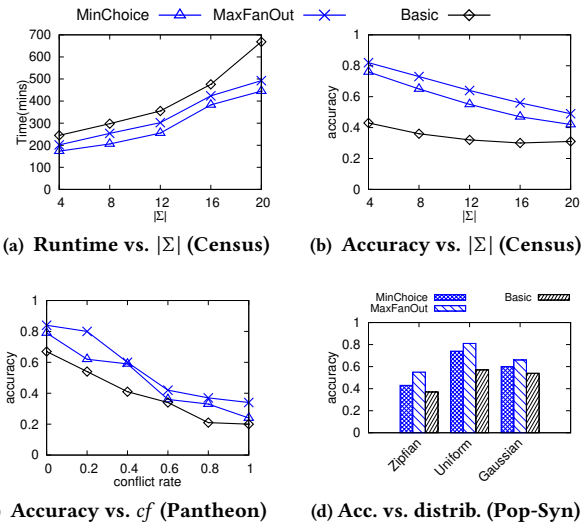
MaxFanOut: We select constraints with maximum overlap with other constraints, i.e, nodes with the maximum number of uncolored neighbors. We preferentially select these constraints due to their high number of interactions, which lead to an increased number of target attributes, and bounds that the relevant tuples must satisfy. This heuristic aims to prune unsatisfiable clusterings early to reduce the number of clustering evaluations downstream.

4 EXPERIMENTS

We evaluate the accuracy and performance of *DIVA*, and compare against three k-anonymization baselines to evaluate the cost of incorporating diversity constraints.

Experimental Setup. We implement *DIVA* using Python 3.6 on a server with 32 Core Intel Xeon 2.2 GHz processor with 32GB RAM. We use three real datasets: (1) *Pantheon* describes individuals on Wikipedia [1]; (2) *Census* from the U.S. Census Bureau describes population data [3]; and (3) *German Credit* describes credit risk according to demographic attributes [3]. We also generate a synthetic population dataset, *Pop-Syn* to specify population characteristics using Synner.io [18]. Table 4 summarizes the dataset characteristics. We implement three notions of diversity via three classes of diversity constraints, namely, minimum frequency, average, and proportional representation from the attribute domain [23]. We found proportion constraints capture the relative distribution in the attribute domain with less sensitivity than average, and run our experiments using proportion constraints. The set of defined constraints, datasets and our source code are available at [2, 19].

Metrics and Parameters. We compute the average runtime over five executions. To quantify accuracy, we seek anonymizations with indistinguishable tuples, and minimize information loss. We use the *discernibility metric*, $disc(R', k)$, which assigns a penalty to each tuple based on the number of tuples that are indistinguishable from it in R' for a given k, reflecting its information loss [4]. We measure the *conflict rate* between a pair of diversity constraints as the number of overlapping relevant tuples, and we extend this definition to a set of diversity constraints. Conflict values range from [0,1], where 0 indicates no overlap. The discernibility metric and conflict rate definitions can be found in [19]. Table 5 lists parameter ranges and default values.



(a) Runtime vs. $|\Sigma|$ (Census) (b) Accuracy vs. $|\Sigma|$ (Census)

(c) Accuracy vs. cf (Pantheon) (d) Acc. vs. distrib. (Pop-Syn)

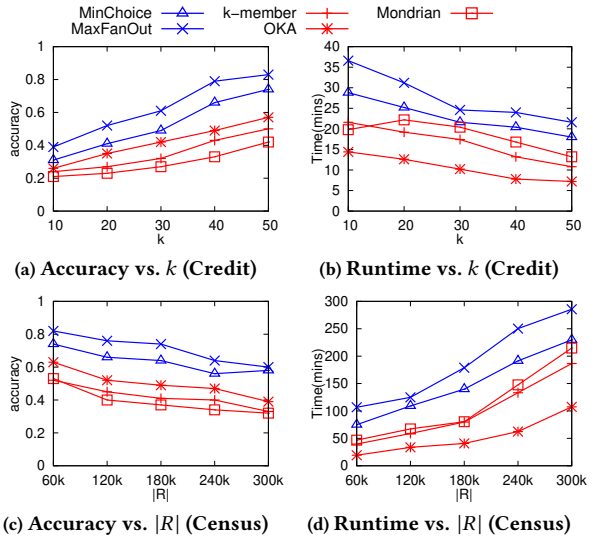
Figure 4: *DIVA* efficiency and effectiveness.

4.1 Accuracy and Performance

Figure 4a and 4b show *DIVA* runtime and accuracy, respectively, as we vary $|\Sigma|$. DIVA-Basic shows exponential growth in runtime since we can assign $O(|R|)$ different clusterings to each constraint. Our selection strategies to restrict clusterings and perform early pruning, MinChoice and MaxFanOut, show linear scale-up. Figure 4b shows as $|\Sigma|$ increases, we see accuracy decline at a linear rate. As new $\sigma \notin \Sigma$ are added, we observe new relevant tuples join existing clusters of relevant tuples from Σ leading to a smaller decline in accuracy. This occurs with multi-attribute constraints that share target attributes with single attribute constraints. The alignment of QI and target attribute values between new and existing tuples influence the rate of decline.

Figure 4c shows *DIVA* accuracy as we vary conflict rate, cf. As expected, accuracy declines for increasing cf, with MaxFanOut and MinChoice outperforming DIVA-Basic by +17% and +9%, respectively. MaxFanOut outperforms MinChoice since targeting constraints with a high number of interactions eliminates unsatisfying clusterings sooner, while satisfying dependent constraints.

To measure accuracy for different data distributions, we generate attribute values according to the Zipfian, uniform, and Gaussian distributions with $|R| = 100k$, $|\Sigma| = 8$. Figure 4d shows that MaxFanOut performs best across all distributions by 8% and 17% over MinChoice and DIVA-Basic, respectively. The target uniform distribution performs best as domain values are spread evenly across the tuples, avoiding contention among a small set of tuples. This conflict occurs more often in the Zipfian case than the Gaussian, leading to lower accuracy.



(a) **Accuracy vs. k (Credit)** (b) **Runtime vs. k (Credit)**

(c) **Accuracy vs. $|R|$ (Census)** (d) **Runtime vs. $|R|$ (Census)**

Figure 5: Comparative study: anonymization baselines.

4.2 Baseline Comparison

Figure 5a and Figure 5b show the comparative accuracy and runtime, respectively, of *DIVA* against three baseline k-anonymization algorithms: k-member [6], OKA [17], and Mondrian [16]. MinChoice and MaxFanOut incur higher runtimes reflecting the cost of computing a *diverse* data instance. We expect this to be acceptable in practice since constraint validation and anonymization is often done offline. As k increases, we expect more tuples to be anonymized leading to higher penalty costs. We note that runtimes decrease for increasing k as more values are suppressed to satisfy k-anonymity on each cluster. This improves the efficiency of consistency checking in the coloring algorithm since we can prune clusterings with size less than k during backtracking.

Figures 5c and 5d show that *DIVA* and the baselines are sensitive to $|R|$. Figure 5c shows that *DIVA* achieves improved accuracy over the baseline algorithms, in addition to satisfying diversity constraints. As new attribute values are introduced, they may not align with existing values in the clusters, requiring further suppression, and decreasing accuracy. In Figure 5d, all techniques incur increased runtimes due to evaluating a larger number of clusters. For *DIVA*, a larger number of tuples and clusters also increases the likelihood of potential conflicts among clusterings.

5 RELATED WORK

Privacy Preserving Data Publishing. Extensions of k-anonymity include *l-diversity*, *t-closeness*, and *(X,Y)-anonymity* with tighter privacy guarantees [11]. *DIVA* is extensible to re-define the clustering criteria according to these privacy semantics. Our empirical study has shown that *DIVA* generates anonymized instances comparable to existing baselines, but also guarantees diversity requirements are satisfied.

Fairness and Diversity. Data sharing of private data has been studied along two primary lines. First, causality reasoning aims to recognize discrimination to achieve algorithmic transparency and fairness. Recent techniques have proposed influence measures to identify correlated attributes [8], and statistical reasoning about discrimination [20]. Secondly, recent work have studied variants of DP to release synthetic data with similar statistical properties to the input data [5], and studying the impact of DP algorithms on equitable resource allocation [21]. Our work is complementary but with a different goal; to publish diverse and anonymized versions of the original data with minimal information loss for applications where statistical summaries, synthetic data, and aggregate queries are inadequate. Recent work by Stoyanovich et. al., study diversity in the set selection problem and introduce diversity constraints to guarantee representation in the selected set [23, 25]. We build upon this work, and are the first to formalize diversity constraints. Our algorithms couple diversity with data anonymization, a problem not considered in existing work.

6 CONCLUSION AND FUTURE WORK

We formalize diversity constraints, and introduce *DIVA*, a DIVersity-driven Anonymization algorithm that computes a privatized data instance satisfying a set of diversity constraints. Our evaluation show the performance benefits of our optimizations, and the overhead of enforcing diversity constraints over baselines. As future work, we intend to study privacy extensions beyond k-anonymity, e.g. randomization algorithms to satisfy both diversity constraints and *Differential privacy (DP)* to provide a higher level of protection for individuals [10]. We also intend to explore a distributed version of the coloring algorithm to improve scalability by satisfying constraints in parallel.

REFERENCES

[1] 2014. *Pantheon Dataset.* https://pantheon.world/
[2] 2020. *DIVA: Extended Evaluation.* https://diva1234567.github.io/DIVA/
[3] 2020. *UCI ML Repository.* https://archive.ics.uci.edu/ml/datasets/
[4] R. Bayardo and R. Agrawal. 2005. Data Privacy through Optimal K-Anonymization. In *ICDE*. 217–228.
[5] V. Bindschaedler, R. Shokri, and C. Gunter. 2017. Plausible Deniability for Privacy-Preserving Data Synthesis. *PVLDB* 10, 5 (2017), 481–492.
[6] J. Byun, A. Kamra, E. Bertino, and N. Li. 2007. Efficient k-anonymization using clustering techniques. In *DASFAA*. 188–200.
[7] F. Chiang and D. Gairola. 2018. InfoClean: Protecting Sensitive Information in Data Cleaning. *Journal of Data and Information Quality* 9, 4 (2018), 22:1–22:26.
[8] A. Datta, S. Sen, and Y. Zick. 2016. Algorithmic Transparency via Quantitative Input Influence: Theory and Experiments with Learning Systems. In *Symposium on Security and Privacy*. 598–617.
[9] M. Drosou, H. Jagadish, E. Pitoura, and J. Stoyanovich. 2017. Diversity in Big Data: A Review. *Big Data* 5, 2 (2017), 73–84.
[10] C. Dwork. 2006. Differential Privacy. In *International Colloquium on Automata, Languages and Programming*. 1–12.
[11] B. Fung, K. Wang, R. Chen, and P. Yu. 2010. Privacy-Preserving Data Publishing: Survey of Recent Developments. *ACM Comput. Surv.* 42, 4 (2010).
[12] B. Grau and E. Kostylev. 2016. Logical Foundations of Privacy-preserving Publishing of Linked Data. In *AAAI*. 943–949.
[13] M. Hay, G. Miklau, D. Jensen, D. Towsley, and C. Li. 2010. Resisting structural reidentification in anonymized social networks. *VLDB J.* 19, 6 (2010), 797–823.
[14] Y. Huang, M. Milani, and F. Chiang. 2018. PACAS: Privacy-Aware, Data Cleaning-as-a-Service. In *International Conference on Big Data*. 1023–1030.
[15] Y. Huang, M. Milani, and F. Chiang. 2020. Privacy-aware data cleaning-as-a-service. *Information Systems* 94 (2020), 101608.
[16] K. LeFevre, D. DeWitt, and R. Ramakrishnan. 2006. Mondrian multidimensional k-anonymity. In *ICDE*. IEEE, 25–25.
[17] J. Lin and M. Wei. 2008. An efficient clustering method for k-anonymization. In *PAIS*. 46–50.
[18] M. Mannino and A. Abouzied. 2019. Is This Real? Generating Synthetic Data That Looks Real. In *UIST*. 549–561.
[19] M. Milani, Y. Huang, and F. Chiang. 2020. Diversifying Anonymized Data with Diversity Constraints. *arXiv preprint arXiv:2007.09141* (2020).
[20] R. Nabi and I. Shpitser. 2018. Fair Inference on Outcomes. In *AAAI*. 1931–1940.
[21] D. Pujol, R. McKenna, S. Kuppam, M. Hay, A. Machanavajjhala, and G. Miklau. 2020. Fair Decision Making Using Privacy-Protected Data. In *FAccT*. 189–199.
[22] P. Samarati. 2001. Protecting respondents identities in microdata release. *TKDE* 13, 6 (2001), 1010–1027.
[23] J. Stoyanovich, K. Yang, and H. Jagadish. 2018. Online set selection with fairness and diversity constraints. In *EDBT*. 241–252.
[24] L. Sweeney. 2002. k-anonymity: A model for protecting privacy. *Int. J. Uncertain. Fuzziness Knowl.-Based Syst.* 10, 05 (2002), 557–570.
[25] K. Yang and J. Stoyanovich. 2017. Measuring Fairness in Ranked Outputs. In *SSDBM*. 22:1–22:6.