Unique Entity Estimation with Application to the Syrian Conflict

Beidi Chen, Anshumali Shrivastava, and Rebecca C. Steorts

Department of Computer Science
Rice University, Houston, TX USA
e-mail: beidi.chen@rice.edu; anshumali@rice.edu

Department of Statistical Science
and Computer Science
Duke University
Durham, NC USA
e-mail: beka@stat.duke.edu

Abstract: Entity resolution identifies and removes duplicate entities in large, noisy databases and has grown in both usage and new developments as a result of increased data availability. Nevertheless, entity resolution has tradeoffs regarding assumptions of the data generation process, error rates, and computational scalability that make it a difficult task for real applications. In this paper, we focus on a related problem of unique entity estimation, which is the task of estimating the unique number of entities and associated standard errors in a data set with duplicate entities. Unique entity estimation shares many fundamental challenges of entity resolution, namely, that the computational cost of all-to-all entity comparisons is intractable for large databases. To circumvent this computational barrier, we propose an efficient (near-linear time) estimation algorithm based on locality sensitive hashing. Our estimator, under realistic assumptions, is unbiased and has provably low variance compared to existing random sampling based approaches. In addition, we empirically show its superiority over the state-of-the-art estimators on three real applications. The motivation for our work is to derive an accurate estimate of the documented, identifiable deaths in the ongoing Syrian conflict. Our methodology, when applied to the Syrian data set, provides an estimate of $191,874 \pm 1772$ documented, identifiable deaths, which is very close to the Human Rights Data Analysis Group (HRDAG) estimate of 191,369. Our work provides an example of challenges and efforts involved in solving a real, noisy challenging problem where modeling assumptions may not hold.

Keywords and phrases: Syrian conflict, entity resolution, clustering, hashing.

1. Introduction

Our work is motivated by a real estimation problem associated with the ongoing conflict in Syria. While deaths are tremendously well documented, it is hard to know how many unique individuals have been killed from conflict-related violence in Syria. Since March 2011, increasing reports of deaths have appeared in both the national and international news. There are many inconsistencies from various media sources, which is inherent due to the data collection process and the fact that reported victims are documented by multiple sources. Thus, our ultimate goal is to determine an accurate number of documented, identifiable deaths (with associated standard errors) because such information may contribute to future transitional justice and accountability measures. For instance, statistical estimates of death counts have been introduced as evidence in national court cases and international tribunals investigating the responsibility of state leaders for crimes against humanity (Grillo, 2016).

The main challenge with reliable death estimation of the Syrian data set is the fact that individuals who are documented as dead are often duplicated in the data sets. To address this challenge, one could employ entity resolution (deduplication or record linkage), which refers to the task of removing duplicated records in noisy datasets that refer to the same entity (Tancredi and Liseo, 2011; Sadinle et al., 2014; Bhattacharya and Getoor, 2006; Baxter et al., 2003; Gutman, Afendulis and Zaslavsky, 2013; Winkler, 2004; McCallum and Wellner, 2004; Deming and Glasser, 1959; Fellegi and Sunter, 1969). Entity resolution is fundamental in many large data processing applications. Informally, let us assume that each entity (records) is a vector in $\mathbb{R}^D$. Then given a data set of M records aggregated from many data sources with possibly numerous duplicated entities perturbed by noise, the task of entity resolution is to identify and remove the duplicate entities. For a review of entity resolution see (Winkler, 2006; Christen, 2012; Liseo and Tancredi, 2013).

One important subtask of entity resolution is estimating the number of unique entities (records) n out of $M > n$ duplicated entities, which we call *unique entity estimation*. Entity resolution is a more difficult problem because it requires one to link each entity to its associated duplicate entities. To obtain high-accuracy entity resolution, the algorithms must at least evaluate a significant amount of pairs for potential duplicates to ensure a link is not missed. Due to this (and to the best of our knowledge), accurate entity resolution algorithms scale quadratically or higher ($> O(M^2)$) making them computationally intractable for large data sets. Reducing the computational cost in entity resolution is known as blocking, which, via deterministic or probabilistic algorithms, places similar records into blocks or bins (Christen, 2012; Steorts et al., 2014). The computational efficiency comes at the cost of missed links and reduced accuracy for entity resolution. Further, it is not clear if we can use these crude but cheap entity resolution sub-routines for unbiased estimation of unique entities with strong statistical guarantees.

The primary focus of this paper is on developing a *unique entity estimation* algorithm that is motivated by the ongoing conflict in Syria and has the following desiderata:

1. The estimation cost should be significantly less than quadratic ($O(M^2)$). In particular, any methodology requiring one to evaluate all pairs for linkage is not suitable. This is crucial for the Syrian data set and other large, noisy data sets (Section 1.3).
2. To ensure accountability regarding estimating the unique number of documented identifiable victims in the Syrian conflict, it is essential to under-

stand the statistical properties of any proposed estimator. Such a requirement eliminates many heuristics and rule-based entity resolution tasks, where the estimates may be very far from the true value.

3. In most real entity resolution tasks, duplicated data can occur with arbitrarily large changes including missing information, which we observe in the Syrian data set, and standard modeling assumptions may not hold due to the noise inherent in the data. Due to this, we prefer not to make strong modeling assumptions regarding the data generation process.

1.1. Related Work for Unique Entity Estimation

The three aforementioned desiderata eliminate all but random sampling-based approaches. In this section, we review them briefly.

To our knowledge, only two random sampling based methodologies satisfy such requirements. Frank (1978) proposed sampling a large enough subgraph to estimate the total number of connected components based on the properties of the sub-sampled subgraph. Also, Chazelle, Rubinfeld and Trevisan (2005) proposed finding connected components with high probability by sampling random vertices and then visiting their associated components using breadth-first search (BFS). One major issue with random sampling is that most sampled pairs are unlikely to be matches (no edge) providing nearly no information, as the underlying graph is generally very sparse in practice. Randomly sampling vertices and running BFS required by Chazelle, Rubinfeld and Trevisan (2005) are very likely to result in singleton vertices because many records are themselves unique in entity resolution data sets. In addition, finding all possible connections of a given vertex would require $O(M)$ query for edges. A query for edges corresponds to the query for actual link between two records. Sub-sampling a sub-graph, as in Frank (1978), of size s requires $O(s^2)$ edge queries to completely observe it. Thus, s should be reasonably small in order to scale. Unfortunately, requiring a small s hurts the variance of the estimator. We show that the accuracy of both aforementioned methodologies is similar to the non-adaptive variant of our estimator which has provably large variance. In addition, we show both theoretically and empirically that the methodologies based on random sampling lead to poor estimators.

While some methods have recently been proposed for accurate estimation of unique records, they belong to the Bayesian literature and have difficulty scaling due to the curse of dimensionality with Markov chain Monte Carlo Steorts, Hall and Fienberg (2014, 2016); Steorts (2015); Sadinle et al. (2014); Tancredi and Liseo (2011); Zanella et al. (2016). The evaluation of the likelihood itself is quadratic. Furthermore, they rely on a strong assumption about the specified generative models for the duplicate records. Given such computational challenges with the current state of the methods in the literature, we take a simple approach, especially given the large and constantly growing data sets that we seek to analyze. We focus on practical methodologies that can easily scale to large data sets with minimal assumptions. Specifically, we propose a unique entity estimation algorithm with sub-quadratic cost, which can be reduced to approximating the number of connected components in a graph with sub-quadratic queries for edges (Section 3.1).

The rest of the paper proceeds as follows. Section 1.2 provides our motivational application from the Syrian conflict and Section 1.3 remarks on the main challenges of the Syrian data set and our proposed methodology. Section 2.1 provides background on variants of locality sensitive hashing (LSH), which is essential to our proposed methodology. Section 3 provides our proposed methodology for unique entity estimation, which is the first formalism of using efficient adaptive LSH on edges to estimate the connected components with sub-quadratic computational time. (An example of our approach is given in section 3.2). More specifically, we draw connections between our methodology and random and adaptive sampling in section 3.3, where we show under realistic assumptions that our estimator is theoretically unbiased and has provably low variance. In addition, in section 3.5, we compare random and adaptive sampling for the Syrian data set, illustrating the strengths of adaptive sampling. In section 3.6, we introduction the variant of LSH used in our paper. Section 3.7 provides our complete algorithm for unique entity estimation. Section 4 provides evaluations of all the related estimation methods on three real data sets from the music and food industries as well as official statistics. Section 5 reports the documented identifiable number of deaths in the Syrian conflict (with a standard error).

1.2. The Syrian Conflict

Thanks to Human Rights Data Analysis Group (HRDAG), we have access to four databases from the Syrian conflict which cover roughly the same period, namely March 2011 – April 2014, namely, the Violation Documentation Centre (VDC), Syrian Center for Statistics and Research (CSR-SY), Syrian Network for Human Rights (SNHR), and Syria Shuhada website (SS). Each database lists a different number of recorded victims killed in the Syrian conflict, along with available identifying information including full Arabic name, date of death, death location, and gender.[1]

Since the above information is collected indirectly, such as through friends and religious leaders, or traditional media resources, it naturally comes with many challenges. The data set has biases, spelling errors, and missing values. In addition, it is well known that there are duplicate entities present in the data sets, making estimation more difficult. The ambiguities in Arabic names make the situation significantly worse as there can be a large textual difference between the full and short names in Arabic. (It is not surprising that the Syrian data set has such biases given that the data is collected in the midst of a conflict).

Such ambiguities and lack of additional information make entity resolution on this data set considerably challenging (Price et al., 2014). Owing to the

[1]These databases include documented identifiable victims and not those who are missing in the conflict, hence, any estimate reported only refers to the data at hand.

significance of the problem, HRDAG has provided labels for a large subset of the data set. More specifically, five different human experts from the HRDAG manually reviewed pairs of records in the four data sets, classifying them as matches if referred to the same entity and non-matches otherwise. *Our first* **goal** *is to accurately estimate the number of unique victims.* Obtaining a match or non-match label of a given record pair may require momentous cost such as manual human supervision or involving sophisticated machine learning. Given that coming up with hand-matched data is a costly process, *our second* **goal** is to provide a proxy, automated mechanism to create labeled data. (More information regarding the Syrian data set can be found in Appendix **??**).

1.3. Challenges and Proposed Solutions

Consider evaluating the Syrian data set using all-to-all records comparisons to remove duplicate entities. With approximately 354,000 records from the Syrian data set, we have around 63 billion pairs (6.3×10^{10}). Therefore, it is impractical to classify all these pairs as matches/non-matches reliably. We cannot expect a few experts (five in our case) to manually label 63 billion pairs. A simple computation of all pairwise similarity (63 billion) takes more than 8 days on a heavyweight machine that can run 56 threads in parallel (28 cores in total). In general, this quadratic computational cost is widely considered infeasible for large data sets. Algorithmic labeling of every pair, even if possible for relatively small datasets, is neither reliable nor efficient. Furthermore, it is hard to understand the statistical properties of algorithmic labelling of pairs. Such challenges, therefore, motivate us to focus on the estimation algorithm with constraints mentioned in Section 1.

Our Contributions: We formalize unique entity estimation as approximating the number of connected components in a graph with sub-quadratic $\ll O(M^2)$ computational time. We then propose a generic methodology that provides an estimate in sample (with standard errors). Our proposal leverages locality sensitive hashing (LSH) in a novel way for the estimation process, with the required computational complexity that is less than quadratic. Our proposed estimator is unbiased and has provably low variance compared to random sampling based approaches. To the best of our knowledge this is the first use of LSH for unique entity estimation in an entity resolution setting. Our unique entity estimation procedure is broadly applicable to many applications, and we illustrate this on three additional real, fully labelled, entity resolution data sets, which include the food industry, the music industry, and an application from official statistics. In the absence of ground truth information, we estimate that the number of documented identifiable deaths for the Syrian conflict is 191,874, with standard deviation of 1,772, reported casualties, which is very close to the 2014 HRDAG estimate of 191,369. This clearly demonstrates the power of our efficient estimator in practice, which does not rely on any strong modeling assumptions. Out of 63 billion possible pairs, our estimator only queries around 450,000 adaptively sampled pairs ($\simeq O(M)$) for labels, yielding a 99.99% reduction. The labelling was done using support vector machines (SVMs) trained on a small number of hand-matched, labeled examples provided by five domain experts. Our work is an example of the efforts required to solve a real noisy challenging problem where modeling assumptions may not hold.

2. Variants of Locality Sensitive Hashing (LSH)

In this section, we first provide a review of LSH and min-wise hashing, which is crucial to our proposed methodology. We then introduce a variant of LSH — Densified One Permutation Hashing (DOPH), which is essential to our proposed algorithm for unique entity estimation in terms of scalability. We first provide a brief literature review of LSH.

2.1. Review of Locality Sensitive Hashing (LSH)

In this section, we first provide a review of locality sensitive hashing and min-wise hashing, which is crucial to our proposed methodology.

Locality sensitive hashing (LSH) is a well-known *probabilistic method* of dimension reduction, which is widely used in computer science and in database engineering as a way of rapidly finding approximate nearest neighbors (Gionis et al., 1999). More recently, locality sensitive hashing has been utilized has a form of blocking in entity resolution, where one tries to achieve scalability and avoid all-to-all record comparisons by placing records into "partitions" or "blocks" either using deterministic or probabilistic methods.

Unlike other conventional forms of dimension reduction or blocking for entity resolution, LSH uses all the features of a record, and can be adjusted to ensure that blocks are manageably small, but then do not allow for further record linkage within blocks. For example, Vatsalan et al. (2014) introduced novel data structures for sorting and fast approximate nearest-neighbor look-up within blocks produced by LSH. Their approach gave a good balance between speed and recall, but their technique is very specific to nearest neighbor search. In other related work, Steorts et al. (2014) proposed clustering-based blocking schemes that are variants on LSH. The first, transitive locality sensitive hashing (TLSH) is based upon the community discovery literature such that *a soft transitivity* (or relaxed form of transitivity) can be imposed across blocks. The second, k-means locality sensitive hashing (KLSH) is based upon the information retrieval literature and clusters similar records into blocks using a vector-space representation and projections (KLSH had been used before in information retrieval but never with entity resolution (Paulevé, Jégou and Amsaleg, 2010)). Steorts et al. (2014) showed that both KLSH and TLSH gave improvements over popular methods in the literature such as traditional blocking, canopies (McCallum, Nigam and Ungar, 2000), and k-nearest neighbors clustering.

There are many variants of LSH and one popular form is min-wise hashing. All LSH methods are defined by a type of similarity and a type of dimension reduction (Broder, 1997a). Recently, Shrivastava and Li (2014a) showed that min-wise hashing based approaches are superior to random projection based approaches when the data is very sparse and feature poor. Furthermore, improvements in computational speed can be obtained by using the recently proposed densification scheme known as densified one permutation hashing (DOPH) (Shrivastava and Li, 2014a,b). Specifically, the authors proposed an efficient substitute for min-wise hashing, which only requires one permutation (or one hash function) for generating many different hash values needed for indexing. In short, the algorithm is linear (or constant) in the tuning parameters, making it very computationally efficient.

2.2. Shingling

In entity resolution tasks, each record can be represented as a string of information. For example, each record in the Syrian data set can be represented as a short *text* description of the person who died in the conflict. In this paper, we use a k-grams based shingle representation, which is the most common representation of text data and naturally gives a set token (or k-grams). That is, each record is treated as a string and is replaced by a "bag" (or "multi-set") of length-k contiguous sub-strings that it contains. Since we will use a k-gram based approach to transform the records, our representation of each record will also be a set, which consists of all the k-contiguous characters occurring in record string. As an illustration, for the record BAKER, TED, we separate it into a 2-gram representation. The resulting set is the following:

BA, AK, KE, ER, RT, TE, ED.

In another example, consider Sammy, Smith, whose 2-gram set representation is

SA, AM, MM, MY, YS, MS, SM, MI, IT, TH.

We now have two records that have been transformed into a 2-gram representation. Thus, for every record (string) we obtain a set $\subset \mathcal{U}$, where the universe $\mathcal{U}$ is the set of all possible k-contiguous characters.

2.3. Locality Sensitive Hashing

In this paper, we leverage LSH, which comes with sound mathematical formalism and guarantees. LSH is widely used in computer science and database engineering as a way of rapidly finding approximate nearest neighbors (Indyk and Motwani, 1998; Gionis et al., 1999). Specifically, the variant of LSH that we utilize is scalable to large databases, and allows for similarity based sampling of entities in less than a quadratic amount of time.

In LSH, a hash function is defined as $y = h(x)$, where y is the *hash code* and $h(\cdot)$ the *hash function.* A *hash table* is a data structure that is composed of *buckets* (not to be confused with blocks), each of which is indexed by a *hash code.* Each reference item x is placed into a bucket $h(x)$.

More precisely, LSH is a family of functions that map vectors to a discrete set, namely, $h : \mathbb{R}^D \to \{1,\ 2, \cdots, M\}$, where M is in finite range. Given this family of functions, similar points (entities) are likely to have the same hash value compared to dissimilar points (entities). The notion of similarity is specified by comparing two vectors of points (entities), x and y. We will denote a general notion of similarity by $\mathrm{SIM}(x, y)$. In this paper, we only require a relaxed version LSH, and we define this below. Formally, a LSH is defined by the following definition below:

Definition 1. *(Locality Sensitive Hashing (LSH)) Let $x_1,\ x_2,\ y_1,\ y_2 \in \mathbb{R}^D$ and suppose h is chosen uniformly from a family $\mathcal{H}$. Given a similarity metric, $SIM(x, y)$, $\mathcal{H}$ is locality sensitive if $SIM(x_1, x_2) \geq Sim(y_2, y_3)$ then $Pr_{\mathcal{H}}(h(x_1) = h(x_2)) \geq Pr_{\mathcal{H}}(h(y_1) = h(y_2))$, where $Pr_{\mathcal{H}}$ is the probability over the uniform sampling of h.*

The above definition is sufficient condition for a family of functions to be LSH. While many popular LSH families satisfy the aforementioned property, we only require this condition for the work described herein. For a complete review of LSH, we refer to Rajaraman and Ullman (2012).

2.4. Minhashing

One of the most popular forms of LSH is minhashing (Broder, 1997b), which has two key properties — a type of similarity and a type of dimension reduction. The type of similarity used is the Jaccard similarity and the type of dimension reduction is known as the minwise hash, which we now define.

Let $\{0, 1\}^D$ denote the set of all binary D dimensional vectors, while $\mathbb{R}^D$ refers to the set of all D dimensional vectors (of records). Note that records can be represented as a binary vector (or set) representation via shingling, BoW, or combining these two methods. More specifically, given two record sets (or equivalently binary vectors) $x, y \in \{0, 1\}^D$, the Jaccard similarity between $x, y \in \{0, 1\}^D$ is

$$\mathcal{J} = \frac{|x \cap y|}{|x \cup y|},$$

where $|\cdot|$ is the cardinality of the set.

More specifically, the minwise hashing family applies a random permutation π, on the given set S, and stores only the minimum value after the permutation mapping, known as the *minhash*. Formally, the minhash is defined as $h_\pi^{min}(S) = \min(\pi(S))$, where $h(\cdot)$ is a hash function.

Given two sets S_1 and S_2, it can be shown by an elementary probability argument that

$$Pr_\pi(h_\pi^{min}(S_1) = h_\pi^{min}(S_2)) = \frac{|S_1 \cap S_2|}{|S_1 \cup S_2|}, \tag{1}$$

where the probability is over uniform sampling of π. It follows from Equation 1 that minhashing is a LSH family for the Jaccard similarity.

Remark: In this paper, we utilize a shingling based approach, and thus, our representation of each record is likely to be very sparse. Moreover, Shrivastava and Li (2014c) showed that minhashing based approaches are superior compared to random projection based approaches for very sparse datasets.

2.4.1. Densified One Permutation Hashing (DOPH)

LSH has been utilized for more than two-decades, where one can use LSH to reduce the computational cost of entity resolution. More specifically, the main idea is to only match records which have the same hash values, known as blocking or indexing. One major issue with LSH is that the step of creating blocks (hash buckets) is expensive because it requires several hash computations (Liang et al., 2014; Steorts et al., 2014). However, it was recently shown that the several minwise hashes of data can be computed in data reading time using the technique of Densified One Permutation Hashing (DOPH). Subsequent works (Shrivastava and Li, 2014a,b) improved the statistical properties of DOPH. (Wang, Shrivastava and Ryu, 2017) illustated that using DOPH one can get significant improvements over LSH, which leads to the fastest approximate near-neighbor search algorithm. In this paper, we use the most recent variant of DOPH, which is significantly faster in practice compared to minwise hashing. Since we use a shingle based representation for textual data, DOPH is ideal for our proposed algorithm because the cost for blocking is the same as the data reading cost, which is about 100 times faster than traditional minwise hashing. Throughout the rest of the paper, when we refer to minwise hashing will refer to the DOPH algorithm for computing minhashes. Further details of LSH and DOPH can be found in the aforementioned papers. In addition, we specify another reason for using LSH as the only blocking mechanism which suits our purpose in section 3.6.4.

3. Unique Entity Estimation

In this section, we provide notation used throughout the rest of the paper and provide an illustrative example. We then propose our estimator, which is unbiased and has provably low variance. In addition, random sampling is a special case of our procedure as explained in section 3.5. Finally, we present our unique entity estimation algorithm in section 3.3.

3.1. Notation

The problem of unique entity estimation can be reduced to approximating the number of connected components in a corresponding graph. Given a data set with size M, we denote the records as

$$R = \{R_i | 1 \leq i \leq M,\ i \in \mathbb{Z}\}.$$

Next, we define

$$Q(R_i, R_j) = \begin{cases} 1, & \text{if } R_i, R_j \text{ refer to the same entity }. \\ 0, & \text{otherwise.} \end{cases}.$$

Let us represent the data set by a graph $G^* = (E, V)$, with vertices E, V. Let vertex V_i correspond to record R_i and vertex V_j correspond to record R_j. Then let edge E_{ij} represent the linkage between records of R_i and R_j (or vertex V_i and V_j). More specifically, we can represent this by the following relationship:

$$V = \{R_i | 1 \leq i \leq M, i \in \mathbb{Z}\}, \text{ and } E = \{(R_i, R_j) | \forall\, 1 \leq i, j \leq M,\ Q(R_i, R_j) = 1\}.$$

3.2. Illustrative Example

In this section, we provide an illustrative example of how six records are mapped to a graph G^*. Consider record 3 (John) and record 5 (Johnathan) which correspond to the same entity (John Schaech). In G^*, there is an edge E_{35} that connect these records, denoted by V_3 and V_5. Now consider records 2, 4, and 6, which all refer to the same entity (Nicholas Cage). In G^*, there are edges E_{24}, E_{26}, and E_{46} that connect these records, denoted by V_2, V_4, and V_6. Observe that each connected component in G^* is a unique entity and also a clique. Therefore, our task is reduced to estimating the number of connected components in G^*.

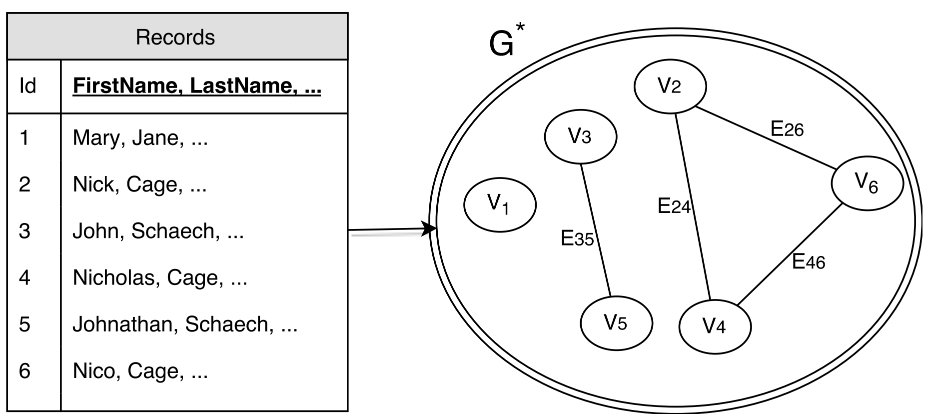

Fig 1: A toy example of mapping records to a graph, where vertices represent records and edges refer to the relation between records.

3.3. Proposed Unique Entity Estimator

In this section, we propose our unique entity estimator and provide assumptions that are necessary for our estimation procedure to be practical (scalable).

Since we do not observe the edges of G^* (the linkage), inferring whether there is an edge between two nodes (or whether two records are linked) can be costly, i.e., $O(M^2)$. Hence, one is constrained to probe a small set $\mathcal{S} \subset V \times V$ with $|\mathcal{S}| \ll O(M^2)$ of pairs and query if they have edges. The aim is to use the information about $\mathcal{S}$ to estimate the total number of connected components accurately. More precisely, given the partial graph $G' = \{V, E'\}$, where $E' = E \cap \mathcal{S}$, one wishes to estimate the connected components n of $G^* = \{V, E\}$.

One key property of our estimation process is that we do not make any modeling assumptions of how duplicate records are generated, and it is not immediately clear how we can obtain unbiased estimation. For sake of simplicity, we first assume the existence of an efficient (sub-quadratic) process that samples a small set (near-linear size) of edges $\mathcal{S}$, such that every edge in the original graph G^* has (reasonably high) probability p of being in $\mathcal{S}$. Thus, set $\mathcal{S}$, even though small, contains p fraction of the actual edges. For sparse graphs, as in the case of duplicate records, such a sampler will be far more efficient than random sampling. Based on this assumption, we will first describe our estimator and its properties. We then show why our assumption about existence of adaptive sampler is practical by providing an efficient sampling process based on LSH (Section 3).

Remark: It is not difficult to see that random sampling is a special case when $p = \frac{|\mathcal{S}|}{O(M^2)}$ which, as we show later, is a very small number for any accurate estimation.

Our proposed estimator and corresponding algorithm obtains the set of vertex pairs (or edges) $\mathcal{S}$ through an efficient (adaptive) sampling process and queries whether there is an edge (linkage) between each pair in $\mathcal{S}$. Respectively, after the ground truth querying, we observe a sub-sampled graph G', consisting of vertices returned by the sampler. Let n'_i be the number of connected component of size i in the observed graph G', i.e., n'_1 is the number of singleton vertices, n'_2 is the number of isolated edges, etc. in G'. It is worth noting that every connected component in G' is a part of some clique (maybe larger) in G^*. Let n^*_i denote the number of connected components (clique) of size i in the original (unobserved) graph G^*.

Observe that under the sampling process, any original connected component, say C^*_i (clique), will be sub-sampled and can appear as some possibly smaller connected component in G'. For example, a singleton set in G^* will remain the same in G'. An isolated edge, on the other hand, can appear as an edge in G' with probability p and as two singleton vertices in G' with probability $1 - p$. A triangle can decompose into three possibilities with probability shown in figure 2. Each of these possibilities provides a linear equation connecting n^*_i to n'_i. These equations up to cliques of size three are

$$\mathbb{E}[n'_3] = n^*_3 \cdot p^2 \cdot (3 - 2p) \tag{2}$$

$$\mathbb{E}[n'_2] = n^*_2 \cdot p + n^*_3 \cdot (3 \cdot (1-p)^2 \cdot p) \tag{3}$$

$$\mathbb{E}[n'_1] = n^*_1 + n^*_2 \cdot (2 \cdot (1-p)) + n^*_3 \cdot (3 \cdot (1-p)^2). \tag{4}$$

Since we observe n'_i, we can solve for the estimator of each n^*_i and compute the number of connected components by summing up all n^*_i.

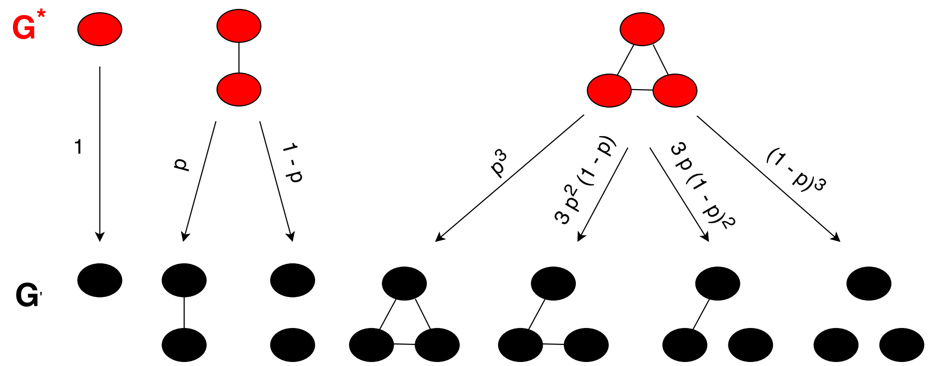

Fig 2: A general example illustrating the transformation and probabilities of connected components from G^* to G'.

Unfortunately, this process quickly becomes combinatorial, and in fact, is at least $\#P$ hard (Provan and Ball, 1983) to compute for cliques of larger sizes. A large clique of size k can appear as many separate connected components and the possibilities of smaller size components it can break into are exponential (Aleksandrov, 1956). Fortunately, we can safely ignore large connected components without significant loss in estimation for two reasons. First, in practical entity resolution tasks, when M is large and contains at least one string-valued feature, it is observed that *most* entities are replicated no more than three or four times. Second, a large clique can only induce large errors if it is broken into many connected components due to undersampling. According to Erdos and Rényi (1960), it will almost surely stay connected if p is high, which is the case with our sampling method.

Assumption: As argued above, we safely assume that the cliques of sizes equal to or larger than 4 in the original graph would retain their structures, i.e., $\forall i \geq 4,\ n^*_i = n'_i$. With this assumption, we can write down the formula for estimating n^*_1, n^*_2, n^*_3 by solving Equations 2–4 as,

$$n^*_3 = \frac{\mathbb{E}[n'_3]}{p^2 \cdot (3-2p)}, \quad n^*_2 = \frac{\mathbb{E}[n'_2] - n^*_3 \cdot (3 \cdot (1-p)^2 \cdot p)}{p} \tag{5}$$

$$n^*_1 = \mathbb{E}[n'_1] - n^*_2 \cdot (2 \cdot (1-p)) - n^*_3 \cdot (3 \cdot (1-p)^2) \tag{6}$$

It directly follows that our estimator, which we call the Locality Sensitive Hashing Estimator (LSHE) for the number of connected components is given by

$$\text{LSHE} = n'_1 + n'_2 \cdot \frac{2p-1}{p} + n'_3 \cdot \frac{1 - 6 \cdot (1-p)^2 \cdot p}{p^2 \cdot (3-2p)} + \sum_{i=4}^{M} n'_i. \tag{7}$$

3.4. Optimality Properties of LSHE

We now prove two properties of our unique entity estimator, namely, that it is unbiased and that is has provably lower variance than random sampling approaches. Here we have assumed independence of sampling. Our sampler relying on LSH, described in Section 3.6, will have even better variance due to favorable correlations. Please see (Spring and Shrivastava, 2017a; Luo and Shrivastava, 2017; Chen, Xu and Shrivastava, 2018; Luo and Shrivastava, 2018) for more details. Those discussions are out of the scope of this paper.

Theorem 1. *Assuming* $\forall i \geq 4,\ n_i^* = n_i'$*, we have*

$$\mathbb{E}[LSHE] = n \quad \textit{unbiased} \tag{8}$$

$$\mathbb{V}[LSHE] = n_3^* \cdot \frac{(p-1)^2 \cdot (3p^2 - p + 1)}{p^2 \cdot (3 - 2p)} + n_2^* \frac{(1-p)}{p} \tag{9}$$

The above estimator is unbiased and the variance is given by Equation 9.

Theorem 2 proves the variance of our estimator is monotonically decreasing with p.

Theorem 2. $\mathbb{V}[LSHE]$ *is monotonically decreasing when* p *increases in range* $(0, 1]$.

The proof of Theorem 2 directly follows Lemma 1, which is immediately given.

Lemma 1. *First order derivative of* $\mathbb{V}[LSHE]$ *is negative when* $p \in (0, 1]$.

Note that when $p = 1$, $\mathbb{V}[\text{LSHE}] = 0$ which means the observed graph G' is exactly the same as G^*. For detailed proofs of unbiasedness and Lemma **??**, see Appendix **??**.

3.5. Adaptive Sampling versus Random Sampling

Before we describe our adaptive sampler, we briefly quantify the advantages of an adaptive sampling over random sampling for the Syrian data set by computing the differences between their variances. Let p be the probability that an edge (correct match) is sampled. On the Syrian data set, our proposed sampler, described in next section, empirically achieves $p = 0.83$, by reporting around 450,000 sampled pairs ($O(M)$) out of the 63 billion possibilities ($O(M^2)$). Substituting this value of p, the corresponding variance can be calculated from Equation 9 as

$$n_3^* \cdot 0.07 + n_2^* \cdot 0.204.$$

Turning to plain random sampling of edges, in order to achieve the same sample size above leads to p as low as $\frac{4.5\times10^5}{6.3\times10^{10}} \simeq 6.9\times10^{-6}$. With such minuscule p, the resulting variance is

$$n_3^* \cdot 6954620166 + n_2^* \cdot 144443.$$

Thus, the variance for random sampling is roughly 7×10^5 times the number of duplicates in the data set and 1×10^{11} the number of triplets in the data set.

In section 4, we illustrate that two other random sampling based algorithms of (Chazelle, Rubinfeld and Trevisan, 2005) and (Frank, 1978) also have poor accuracy compared to our proposed estimator. The poor performance of random sampling is not surprising from a theoretical perspective, and illustrates a major weakness empirically for the task of unique entity estimation with sparse graphs, where adaptive sampling is significantly advantageous.

3.6. The Missing Ingredient: (K,L)-LSH Algorithm

Our proposed methodology, for unique entity estimation, assumes that we have an efficient algorithm that adaptively samples a set of record pairs, in sub-quadratic time. In this section, we argue that using a variant of LSH (Section 2.1) we can construct such an efficient sampler.

As already noted, we do not make any modeling assumptions on the generation process of the duplicate records. Also, we cannot assume that there is a fixed similarity threshold, because in real datasets duplicates can have arbitrarily large similarity. Instead, we rely on the observation that record pairs with high similarity have a higher chance of being duplicate records. That is, we assume that when two entities R_i and R_j are similar in their attributes, it is more likely that they refer to the same entities (Christen, 2012).[2] We note that this probabilistic observation is the weakest possible assumption, and almost always true for entity resolution tasks because linking records by a similarity score is one simple way of approaching entity resolution (Christen, 2012; Winkler, 2006; Fellegi and Sunter, 1969).

The similarity between entities (records) naturally gives us a notion of adaptiveness. One simple adaptive approach is to sample records pairs with probability proportional to their similarity. However, as a prerequisite for such sampling, we must compute all the pairwise similarities and associated probability values with every edge. Computing such a pairwise similarity score is a quadratic operation ($O(M^2)$) and is intractable for large datasets. Fortunately, recent work has shown that (Spring and Shrivastava, 2017b,a; Luo and Shrivastava, 2017; Chen, Xu and Shrivastava, 2018; Luo and Shrivastava, 2018) it is possible to sample pairs adaptively in proportion to the similarity in provably sub-quadratic time using LSH, which we describe in the next section.

3.6.1. (K,L)-LSH Algorithm and Sub-quadratic Adaptive Sampling

We leverage a very recent observation associated with the traditional (K, L) parameterized LSH algorithm. The (K, L) parameterized LSH algorithm is a popular similarity search algorithm, which given a query q, retrieves element x from a preprocessed data set in sub-linear time ($O(KL) \ll M$) with probability

[2] The similarity metric that we use to compare sets of record strings is the Jaccard similarity.

$1-(1-\mathcal{J}(q,\ x)^K)^L$. Here, J denotes the Jaccard similarity between the query and the retrieved data vector x. Our proposed method leverages this (K, L)-parameterized LSH Algorithm, and we briefly describe the algorithm in this section. For complete details refer to (Andoni and Indyk, 2004).

Before we proceed, we define hash maps and keys. We use hash maps, where every integer (or key) is associated with a bucket (or a list) of records. In a hash map, searching for the bucket corresponding to a key is a constant time operation. Please refer to algorithms literature (Rajaraman and Ullman, 2012) for details on hashing and its computational complexity. Our algorithm will require several hash maps, L of them, where a record R_i is associated with a unique bucket in every hash map. The key corresponding to this bucket is determined by minwise hashes of the record R_i. We encourage readers to refer to (Andoni and Indyk, 2004) for implementation details.

More precisely, let h_{ij}, $i = \{1,\ 2, ...,\ L\}$ and $j = \{1,\ 2, ...,\ K\}$ be $K \times L$ minwise hash functions (Equation 1) with each minwise hash function formed by independently choosing the underlying permutation π. Next, we construct L meta-hash functions (or the keys) $H_i = \{h_{i,1}, h_{i,2}, ..., h_{i,K}\}$, where each of the H_i's is formed by combining K different minwise hash functions. For this variant of the algorithm, we need a total of $K \times L$ functions. With such L meta-hash functions, the algorithm has two main phases, namely the data pre-processing and the sampling pairs phases, which we outline below.

- **Data Preprocessing Phase:** We create L different hash maps (or hash tables), where every hash values maps to a bucket of elements. For every record R_i in the dataset, we insert R_j in the bucket associated with the key $H_i(R_j)$, in hash map $i = \{1,\ 2, ...,\ L\}$. To assign K-tuples H_i (meta-hash) to a number in a fixed range, we use some universal random mapping function to the desired address range. See (Andoni and Indyk, 2004; Wang, Shrivastava and Ryu, 2017) for details.
- **Sample Pair Reporting:** For every record R_j in the dataset and from each table i, we obtain all the elements in the bucket associated with key $H_i(R_j)$, where $i = \{1,\ 2, ...,\ L\}$. We then take the union of the L buckets obtained from the L hash tables, and denote this (aggregated) set by A. We finally, report pairs of records (R_i, R_j), where $R \in A$.

Theorem 3. *The (K,L)-LSH Algorithm reports a pair $(R_i,\ R_j)$ with probability $1-(1-\mathcal{J}(R_i,\ R_j)^K)^L$, where $\mathcal{J}(R_i,\ R_j)$ is the Jaccard Similarity between record pairs $(R_i,\ R_j)$.*

Proof: Since all the minwise hashes are independent due to an independent sampling of permutations, the probability that both R_i and R_j belong to the same bucket in any hash table i is $\mathcal{J}(R_i,\ R_j)^K$. Note from equation 1, each meta-hash agreement has probability $\mathcal{J}(R_i,\ R_j)$. Therefore, the probability that pair $(R_i,\ R_j)$ is missed by all the L tables is precisely $(1-\mathcal{J}(R_i,\ R_j)^K)^L$, and thus, the required probability of successful retreival is the complement.

The probabilistic expression $1-(1-\mathcal{J}(R_i,\ R_j)^K)^L$ is a monotonic function of the underlying similarity $Sim(q, y)$ associated with the LSH. In particular, higher similarity pairs have more chance of being retrieved. Thus, LSH provides the required sampling that is adaptive in similarity and is sub-quadratic in running time.

3.6.2. Computational Complexity

The computational complexity for sampling with M records is $O(MKL)$. The procedure requires computing KL minwise hashes for each record. This step is followed by adding every record to L hash tables. Finally, for each record, we aggregate L buckets to form sample pairs. The result of monotonicity and adaptivity of the samples applies to any value of K and L. We choose $O(K \times L) \ll O(M)$ such that we are able to get samples in sub-quadratic time. We further tune K and L using cross-validation to limit the size of our samples. In section 5.3, we evaluate the effect of varying K and L in terms of the recall and reduction ratio. (For a review of the recall and reduction ratio, we refer to (Christen, 2012).) We address the precision at the very end of our experimental procedure to ensure that the recall, reduction ratio, and precision of our proposed unique entity estimation procedure are all as close to 1 as possible while ensuring that the entire algorithm is computationally efficient. For example, on the Syrian data set, we can generate 450,000 samples in less than 127 sec with an adaptive sampling probability (recall) p as high as 0.83. (Note: the preprocessing is of the order of data loading cost using the (K,L)-LSH Algorithm). On the other hand, computing all pairwise similarities (63 billion) takes more than 8 days on the same machine with 28 cores capable of running 56 threads in parallel. We refer to (Sadosky et al., 2015) regarding specific comparisons of traditional and advanced blocking methods. Specifically, figures 1–3 illustrate variants of blocking, which perform extremely poorly on the Syrian data set for two reasons. The first is that the recall and the precision are both extremely low for entity resolution to be practical. The second reason is that under further inspection the blocks sizes are too large to manage for entity resolution problems at scale. Hence, our focus in this paper is one the variant that we find works the best under standard entity resolution evaluation metrics. Next, we describe how this LSH sampler is related to the adaptive sampler described earlier in Section 3.3.

3.6.3. Underlying Assumptions and Connections with p

Recall that we can efficiently sample record pairs $R_i,\ R_j$ with probability $1 - (1 - J(R_i,\ R_j)^K)^L$. Since we are not making any modeling assumptions, we cannot directly link this probability to p, the probability of sampling the right duplicated pair (or linked entities) as required by our estimator LSHE. In the absence of any knowledge, we can get the estimate of p using a small set of labeled linked pairs $\mathcal{L}$. Specifically, we we can estimate the value of p by counting the fraction of matched pairs (true edges) from $\mathcal{L}$ reported by the sampling process.

Note that in practice there is no similarity threshold θ that guarantees that two record pairs are duplicate records. That is, it is difficult in practice to know a fixed θ where $\mathcal{J}(R_i,\ R_j) \geq \theta$ ensures that R_i and R_j are the same entities. However, the weakest possible and reasonable assumption is that high similarity pairs (textual similarity of records) should have higher chances of being duplicate records than lower similarity pairs.

Formally, this assumption implies that there exists a monotonic function f of similarity $\mathcal{J}(R_i,\ R_j)$ such that the probability of any $R_i,\ R_j$ being a duplicate record is given by $f(\mathcal{J}(R_i,\ R_j))$. Since our sampling probability $1 - (1 - \mathcal{J}(R_i,\ R_j)^K)^L$ is also a monotonic function of $\mathcal{J}(R_i,\ R_j)$, we can also write

$$f(\mathcal{J}(R_i,\ R_j)) = g(1 - (1 - \mathcal{J}(R_i,\ R_j)^K)^L),$$

where g is f composed with h^{-1} which is the inverse of $h(x) = 1 - (1 - x^K)^L$. Unfortunately, we do not know the form of f or g.

Instead of deriving g (or f), which requires additional implicit assumptions on the form of the functions, our process estimates p directly. In particular, the estimated value of p is a data dependent mean-field approximation of g, or rather,

$$p = \mathbb{E}[g(1 - (1 - \mathcal{J}(R_i,\ R_j)^K)^L)].$$

Crucially, our estimation procedure does not require any modeling assumptions regarding the generation process of the duplicate records, which is significant for noisy data sets, where such assumptions typically break.

3.6.4. Why LSH?

Although there are several rule-based blocking methodologies, LSH is the only one that is also a random adaptive sampler. In particular, consider a rule-based blocking mechanism, for example on the Syrian data set, which might block on the date of death feature. Such blocking could be a very reasonable strategy for finding candidate pairs. Note that it is still very likely that duplicate records can have different dates of death because the information could be different or misrepresented. In addition, such a blocking method is deterministic, and different independent runs of the blocking algorithm will report the same set of pairs. Even if we find reasonable candidates, we cannot up-sample the linked records to get an unbiased estimate. There will be a systematic bias in the estimates, which does not have any reasonable correction. In fact, random sampling to our knowledge is the only known choice in the existing literature for an unbiased estimation procedure; however, as already mentioned, random uninformative sampling is likely to be very inaccurate.

LSH, on the other hand, can also be used as a blocking mechanism (Steorts et al., 2014). It is, however, more than just a blocking scheme; it is a provably adaptive sampler. Due to randomness in the blocking, different runs of sampler lead to different candidates, unlike deterministic blocking. We can also average over multiple runs to even increase the concentration of our estimates. The adaptive sampling view of LSH has come to light very recently (Spring and Shrivastava, 2017b,a; Luo and Shrivastava, 2017; Chen, Xu and Shrivastava, 2018; Luo and Shrivastava, 2018). With adaptive sampling, we get much sharper unbiased estimators than the random sampling approach. To our knowledge, this is the first study of LSH sampling for unique entity estimation.

3.7. Putting it all Together: Scalable Unique Entity Estimation

We now describe our scalable unique entity estimation algorithm. As mentioned earlier, assume that we have a data set that contains a text representation of the M records. Suppose that we have a reasonably sized, manually labeled training set $\mathcal{T}$. We will denote the set of sampled pairs of records given by our sampling process as $\mathcal{S}$. Note, each element of $\mathcal{S}$ is a pair. Then our scalable entity resolution algorithm consists of three main steps, with the total computational complexity $O(ML + KL + |\mathcal{S}| + |\mathcal{T}|)$. In our case, we will always have $|\mathcal{S}| \ll O(M^2)$ and $KL \ll M$ (in fact, L will be a small constant), which ensures that the total cost is strictly sub-quadratic. The complete procedure is summarized in Algorithm 1.

1. **Adaptively Sample Record Pairs ($O(ML)$):** We regard each record R_i as a short string and replace it by an "n-grams" based representation. Then one computes $K \times L$ minwise hashes of each corresponding string. This can be done in a computationally efficient manner using the DOPH algorithm, which is done in data reading time. Next, once these hashes are obtained, one applies the sampling algorithm described in section 3 in order to generate a large enough sample set, which we denote by $\mathcal{S}$. For each record, the sampling step requires exactly L hash table queries, which are themselves $O(1)$ memory lookups. Therefore, the computational complexity of this step is $O(ML + KL)$.
2. **Query each Sample Pairs:** Given the set of sampled pairs of records $\mathcal{S}$ from Step 1, for every pair of records in $\mathcal{S}$, we query whether these record pairs are a match or non-match. This step requires, $O(|S|)$, queries for the true labels. Here, one can use manually labeled data if it exists. In the absence of manually labeled data, we can also use a supervised algorithm, such as support vector machines or random forests, that is trained on the manually labeled set $\mathcal{T}$ (Section 5).
   (a) **Estimate p:** Given the sampled set of record pairs $\mathcal{S}$, we need to know the value of p, the probability that any given correct pair is sampled. To do so, we use the fraction of true pairs sampled from the labeled training set $\mathcal{T}$. The sampling probability p can be estimated by computing the fraction of the matched pairs of training set records $\mathcal{T}_{match}$ appearing in $\mathcal{S}$. That is, we estimate p (unbiasedly) by

$$p = \frac{|\mathcal{T}_{match} \cap \mathcal{S}|}{|\mathcal{T}_{match}|}.$$

Algorithm 1 LSH-Based Unique Entity Estimation Algorithm

Input: Records R, Labeled Set $\mathcal{T}$, Sample Size m
Output: $LSHE$
$\mathcal{S} = LSHSampling(R)$ (Section 3.6.1)
Get $\mathcal{T}_{match}$ be the linked pairs (duplicate entities) in $\mathcal{T}$
$p = \frac{|\mathcal{T}_{match} \cap \mathcal{S}||}{|\mathcal{T}_{match}|}$
Query every pair in $\mathcal{S}$ for match/mismatch (get actual labels). (Graph G')
$n'_1, n'_2, n'_3 ... n'_M = Traverse(G')$
$LSHE = Equation\ 7\ (p\ ,\ n'_1, n'_2, n'_3 ... n'_M)$

Fig 3: Overview of our proposed unique entity estimation algorithm.

If T is stored in a dictionary, then this step can be done on the fly while generating samples. It only costs $O(\mathcal{T})$ extra work to create the dictionary.

(b) **Count Different Connected Components in G' ($O(M + |\mathcal{S}|)$):** The resulting matched sampled pairs, after querying every sample for actual (or inferred) labels, form the edges of G'. We now have complete information about our sampled graph G'. We can now traverse G' and count all sizes of connected components in G' to obtain n'_1, n'_2, n'_3 and so on. Traversing the graph has computational complexity $O(M + |\mathcal{S}|)$ time using Breadth First Search (BFS).

3. **Estimate the Number of Connected Components in G^* ($O(1)$):** Given the values of p, n'_1, n'_2, and n'_3 we use equation 7 to compute the unique entity estimator LSHE.

4. Experiments

We evaluate the effectiveness of our proposed methodology on the Syrian data set and three additional real data sets, where the Syrian data set is only partially labeled, while the other three data sets are fully labeled. We first perform evaluations and comparisons on the three fully labeled data sets, and then give an estimate of the documented number of identifiable victims for the Syrian data set.

- **Restaurant**: The **Restaurant** data set contains 864 restaurant records collected from Fodor's and Zagat's restaurant guides.[3] There are a total of 112 duplicate records. Attribute information contains name, address, city, and cuisine.
- **CD**: The **CD** data set that includes 9,763 CDs randomly extracted from freeDB.[4] There are a total of 299 duplicate records. Attribute informa-

[3] Originally provided by Sheila Tejada, downloaded from http://www.cs.utexas.edu/users/ml/riddle/data.html.

[4] https://hpi.de/naumann/projects/repeatability/datasets/cd-datasets.html.

DBname	Domain	Size	# Matching Pairs	# Attributes	# Entities
Restaurants	Restaurant Guide	864	112	4	752
CD	Music CDs	9,763	299	106	9,508
Voter	Registration Info	324,074	70,359	6	255,447
Syria	Death Records	354,996	N/A	6	N/A

Table 1: We present five important features of the four data sets. **Domain** reflects the variety of the data type we used in the experiments. **Size** is the number of total records respectively. **# Matching Pairs** shows how many pair of records point to the same entity in each data set. **# Attributes** represents the dimensionality of individual record. **# Entities** is the number of unique records.

tion consists of 106 total features such as artist name, title, genre, among others.

- **Voter**: The **Voter** data has been scraped and collected by (Christen, 2014) beginning in October 2011. We work with a subset of this data set containing 324,074 records. There are a total of 68,627 duplicate records. Attribute information contains personal information on voters from North Carolina including full name, age, gender, race, ethnicity, address, zip code, birth place, and phone number.
- **Syria**: The **Syria** data set comprises data from the Syrian conflict, which covers the same time period, namely, March 2011 – April 2014. This data set is not publicly available and was provided by HRDAG. The respective data sets come from the Violation Documentation Centre (VDC), Syrian Center for Statistics and Research (CSR-SY), Syrian Network for Human Rights (SNHR), and Syria Shuhada website (SS). Each database lists a different number of recorded victims killed in the Syrian conflict, along with available identifying information including full Arabic name, date of death, death location, and gender.[5]

The above datasets cover a wide spectrum of different varieties observed in practice. For each data set, we report summary information in Table 1.

[5]These databases include documented identifiable victims and not those who are missing in the conflict. Hence, any estimate reported only refers to the data at hand.

Id	First Name	Last Name	Gender	Date of Death	Governorate	Location
1	بنيز	ىيحي	F	2011-10-23	Homs	ةرهاقلا عراشةرهاقلا عراش
2	بنيز	ييحي	F	2011-10-23	Homs	عراشعراش
3	بنيز		F	2011-10-25	Homs	ةميدقلا صمح

Fig 4: We show several death records in Syrian dataset from VDC, which allows for public access to some of the data. All of the three records belong to the same entity, labeled by human experts. Record 1 and 2 are similar in all attributes while Record 1 and 3 are very different. Due to the variation in the data, records that are very similar are likely to be linked as the same entity, however, it is more difficult to make decisions when records show differences, such as record 1 and 3. This illustrates some of the limitations from deterministic blocking methods discussed in Section 3.6.4.

4.1. Evaluation Settings

In this section, we outline our evaluation settings. We denote Algorithm 1 as the LSH Estimator (LSHE). We make comparisons to the non-adaptive variant of our estimator (PRSE), where we use plain random sampling (instead of adaptive sampling). This baseline uses the same procedure as our proposed LSHE, except that the sampling is done uniformly. A comparison with PRSE quantifies the advantages of the proposed adaptive sampling over random sampling. In addition, we implemented the two other known sampling methods, for connected component estimation, proposed in (Frank, 1978) and (Chazelle, Rubinfeld and Trevisan, 2005). For convenience, we denote them as Random Sub-Graph based Estimator (RSGE), and BFS on Random Vertex based Estimator (BFSE) respectively. Since the algorithms are based on sampling (adaptive or random), to ensure fairness, we fix a budget m as the number of pairs of vertices considered by the algorithm. Note that any query for an edge is a part of the budget. If the fixed budget is exhausted, then we stop the sampling process and use the corresponding estimate, using all the information available.

We briefly describe the implementation details of the four considered estimators below:

1. **LSHE:** In our proposed algorithm, we use the (K, L) parameterized LSH algorithm to generate samples of record pairs using Algorithm 3, where recall K and L control the resulting sample size (section 5.3). Given K, L as an input to Algorithm 1, we use the sample size as the value of the fixed budget m. Table 2 gives different sample budget sizes (with the corresponding K and L) and corresponding values of p for selected samples in three real data sets.
2. **PRSE:** For a fair comparison, in this algorithm, we randomly sample the same number of record pairs used by LSHE. We then perform the same estimation process as LSHE but instead use $p = \dfrac{2m}{M(M-1)}$, which corresponds to the random sampling probability to get the same number

of samples, which is m.

3. **RSGE (Frank, 1978):** This algorithm requires performing breadth first search (BFS) on each randomly selected vertices. BFS requires knowing all edges (neighbors) of a node for the next step, which requires $M-1$ edge queries. To ensure the fixed budget m, we end the traversal when the number of distinct edge queries reaches the fixed budget m.
4. **BFSE (Chazelle, Rubinfeld and Trevisan, 2005):** This algorithm samples a subgraph and observes it completely. This requires labeling all the pairs of records in the sampled sub-graph. To ensure same budget m, the sampled sub-graph has approximately $\sqrt{2m}$ vertices.

Remark: To the best of our knowledge there have been no experimental evaluations of the two algorithms of (Frank, 1978) and (Chazelle, Rubinfeld and Trevisan, 2005) in the literature. Hence, our results could be of independent interest in themselves.

We compute the relative error (RE), calculated as

$$\mathrm{RE} = \frac{|\mathrm{LSHE} - n|}{n},$$

for each of the estimators, for different values of the budget m. We plot the RE for each of the estimators, over a range of values of m, summarizing the results in figure 5.

All the estimators require querying pairs of records compared to labeled ground truth data for whether they are a match or a non-match. As already mentioned, in the absence of full labeled ground truth data, we can use a supervised classifiers such as SVMs as a proxy, assuming at least some small amount of labeled data exists. By training an SVM, we can use this as a proxy for labeled data as well. We use such a proxy in the Syrian data set because we are not able to query every pair of records to determine whether they are true duplicates or not.

We start with the three data sets where fully labelled ground truth data exists. For LSHE, we compute the estimation accuracy using both the supervised SVM (Section 5) as well as using the fully labelled ground truth data. The difference in these two numbers quantifies the loss in estimation accuracy due to the use of the proxy SVM prediction instead of using ground truth labeled data. In our use of SVMs, we take less than 0.01% of the total number of the possible record pairs as the training set.

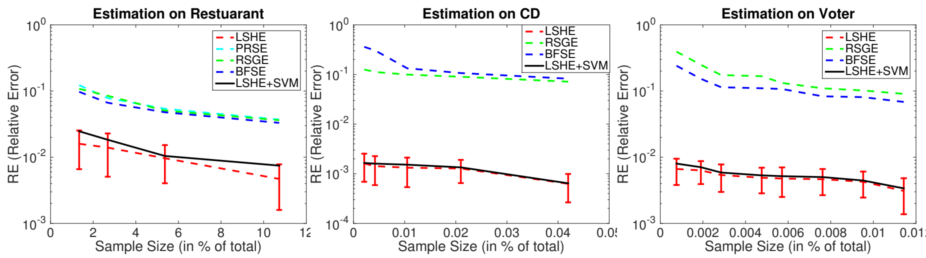

Fig 5: The dashed lines show the RE of the four estimators on the three real data sets, where the y-axis is on the log-scale. Observe that LSHE outperforms all other three estimators in one to two orders of magnitude. The standard deviation of the RE for LSHE is also shown in the plots with the red error bars, which is with respect to randomization of hash functions. In particular, the PRSE performs unreliable estimation on the CD and Voter data sets. The dashed and solid black lines represent RE of LSHE using ground truth labels and a SVM classifier (y-axis is on the log scale). We discuss the LSHE + SVM estimator in section 5 (solid black line).

4.2. Evaluation Results

In this section, we summarize our results regarding the aforementioned evaluation metrics by varying the sample size m on the three real data sets (see figure 5).[6] We notice that for the CD and Voter data sets, we cannot obtain any reliable estimate (for any sample size) using PRSE. Recall that plain random sampling almost always samples pairs of records that correspond to non-matches. Thus, it is not surprising that this method is unreliable because sampling random pairs is unlikely to result in a duplicate pair for entity resolution tasks. Even with repeated trials, there are no edges in the specified sampled pairs of records, leading to an undefined value of p. This phenomenon is a common problem in random sampling estimators over sparse graphs. Almost all the sampled nodes are singletons. Subsampling a small sub-graph leads to a graph with most singleton nodes, which leads to a poor accuracy of BFSE. Thus, it is expected that random sampling will perform poorly. Unfortunately, there is no other baseline for unbiased estimation of the number of unique entities.

From figure 5 observe that the RE for proposed estimator LSHE is approximately one to two orders of magnitude lower than the other considered methods, where the y-axis is on the log-scale. Undoubtedly, our proposed estimator LSHE consistently leads to significantly lower RE (lower error rates) than the other three estimators. This is not surprising from the analysis shown in section 3.5. The variance of random sampling based methodologies will be significantly higher.

[6]For using the fasthash package for unique entity estimation, please see our reproducible code with a tutorial that corresponds with our paper.

Taking a closer look at LSHE, we notice that we are able to efficiently generate samples with very high values of p (see Table 2). In addition, we can clearly see that LSHE achieves high accuracy with very few samples. For example, for the CD data set, with a sample size less than 0.05% of the total possible pairs of records of the entire data set, LSHE achieves 0.0006 RE. Similarly, for the Voter data set, with a sample size less than 0.012% of the total possible pairs of records of the entire data set, LSHE achieves 0.003 RE.

Also, note the small values of K and L parameters required to achieve the corresponding sample size. K and L affect the running time, and small values $KL \ll O(M^2)$ indicate significant computational savings as argued in section 3.6.2

As mentioned earlier, we also evaluate the effect of using SVM prediction as a proxy for actual labels with our LSHE. The dotted plot shows those results. We remark on the results for LSHE + SVM in section 5.

	Restaurant				**CD**				**Voter**			
Size	1.0	2.5	5.0	10	0.005	0.01	0.02	0.04	0.002	0.006	0.009	0.013
p	0.42	0.54	0.65	0.82	0.72	0.74	0.82	0.92	0.62	0.72	0.76	0.82
K	1	1	1	1	1	1	1	1	4	4	4	4
L	4	8	12	20	5	6	8	14	25	32	35	40

Table 2: We illustrate part of the sample sizes (in % in TOTAL) for different sets of samples generated by Min-Wise Hashing and their corresponding p in all three data sets.

5. Documented Identifiable Deaths in the Syrian Conflict

In this section, we describe how we estimate the number of documented identifiable deaths for the Syrian data set. As noted before, we do not have ground truth labels for all record pairs, but the data set was partially labeled with 40,000 record pairs (out of 63 billion). We propose an alternative (automatic) method of labeling the sample pairs, which is also needed by our proposed estimation algorithm. More specifically, using the partially labeled pairs, we train an SVM. In fact, other supervised methods could be considered here, such as random forests, Bayesian Adaptive Regression Trees (BART), among others, however, given that SVMs perform very well, we omit such comparisons as we expect the results to be similar if not worse.

To train the SVM, we take every record pair and generate k-grams representation for each record. Then we spilt the partially labeled data into training and testing sets, respectively. Each training and testing set contains a pair of records $x_k = [R_i,\ R_j]$. In addition, we can use a binary label indicating whether the record pair is a match or non-match. That is, we can write the data as $\{x_k = [R_i, R_j], y_k\}$ as the set difference of the k-grams of the strings of pairs of records R_i and R_j, respectively. Observe that $y_k = 1$ if the R_i and R_j is labelled as match and $y_k = -1$ otherwise. Next, we tune the SVM hyper-parameters using 5-fold cross-validation, and we find the accuracy of SVM on the testing set was 99.9%. With a precision as high a 0.99, we can reliably query an SVM and now treat this as an expert label.

To understand the effect of using SVM prediction as a proxy to label queries in our proposed unique entity estimation algorithm, we return to observing the behavior in figure 5. We treat the LSHE estimator on the other three real datasets as our baselineand compare to LHSE with the SVM component, where the SVM prediction replaces the querying process (LSHE +SVM). Observe in figure 5, that the plot for LSH (solid black line) and LSH+SVM (dotted black line) overlap indicating a negligible loss in performance. This overlap is expected given the high accuracy (high precision) of the SVM classifier.

5.1. Running Time

We briefly highlight the speed of the sampling process since it could be used for on the fly or online unique entity estimation. The total running time for producing 450,000 sampled pairs (out of a possible 63 billion) used for the LSH sampler (Section 3.6.1) with $K = 15$ and $L = 10$ is 127 seconds. The preprocessing cost is included in the 127 seconds. The preprocessing is of the order of data loading cost using DOPH. (For further details on the benchmarking performance of DOPH compared with other LSH methods, please see (Wang, Shrivastava and Ryu, 2017)). On the other hand, it will take approximately take 8 days to compute all pairwise similarities across the 354,996 Syrian records. Computing the pairwise similarities is just the first step for any known adaptive sampling over pairs based on similarity assuming that we do not use the LSH sampler. (Note: there are other ways of blocking (Christen, 2012; Sadosky et al., 2015), however as mentioned in Section 3.6.4 they are mostly deterministic (or rule-based) and do not provide an estimate of the unique entities.

5.2. Unique Number of Documented Identifiable Victims

In the Syrian dataset, with 354,996 records and possibly 63 billion (6.3×10^{10}) pairs, our motivating goal was to estimate the unique number of documented identifiable victims. Specifically, in our final estimate, we use 452,728 sampled pairs that are given by LSHE+SVM ($K = 15$, $L = 10$) which has approximately $p = 0.83$ based on the subset of labeled pairs. The sample size was chosen to balance the computational runtime and the value of p. Specifically, one wants high values of p (for a resulting low variance of our estimate) and, to balance running time, we limit the sample size to be around the total number of records $O(M)$, to ensure a near linear time algorithm. (Such settings are determined by the application, but as we have demonstrated they work for a variety of real entity resolution data sets). We chose the SVM as our classifier to label the matches and non-matches. The final unique number of documented identifiable victims in the Syrian data set was estimated to be 191,874±1772, very close to the 191,369 documented identifiable deaths reported by HRDAG 2014, where their process is described in Appendix **??**.

5.3. Effects of L, K, on sample size and p

In this section, we discuss the sensitivity of our proposed method as we vary the choice of L, K, the sample size M, and p.

We want both $KL \ll M$ as well as the number of samples to be $\ll M^2$, for the process to be truly sub-quadratic. For accuracy, we want high values of p, because the variance is monotonic in p, which is also the recall of true labeled pairs. Thus, there is a natural trade-off. If we sample more, we get high p but more computations.

K and L are the basic parameters of our sampler (Section 3.6.1), which provide a tradeoff between the computationally complexity and accuracy. A large value of K makes the buckets sparse exponentially), and thus, fewer pairs of records are sampled from each table. A large value of L increases the repetition of hash tables (linearly), which increases the sample size. As already argued, the computational cost is $O(MKL)$.

To understand the behavior of K, L, p, and the computational cost, we perform a set of experiments on the Syrian dataset. We use n-gram of 2—5, we vary L from 5–100 by steps of 5 and K takes values 15,18,20,23,25,28,30,32,35. For all these combinations, we then plot the recall (also the value of p) and the reduction ratio (RR), which is the percentage of computational savings. A 99% reduction ratio means that the original space has been reduced to only having to look at a only 1% of total sampled pairs. Figure 6 shows the tradeoffs between reduction ratio and recall (or value of p). Every dot in the figure is one whole experiment.

Regardless of the n-gram variation from 2–5, the recall and reduction ratio (RR) are close to 1 as illustrated in figure 6. We see that an n-gram of 3 overall is most stable in having a recall and RR close to 0.99. We observe that $K = 15$ and $L = 10$ gives a high recall of around 83% with less than half a million pairs (out of 63 billion possible) to evaluate ($RR \geq 0.99999$).

6. Discussion

Motivated by three real entity resolution tasks and the ongoing Syrian conflict, we have proposed a general, scalable algorithm for unique entity estimation. Our proposed method is an adaptive LSH on the edges of a graph, which in turn estimates the connected components in sub-quadratic time. Our estimator is unbiased and has provably low variance in contrast to other such estimators for unique entity estimation in the literature. In experimental results, it outperforms other estimators in the literature on three real entity resolution data sets. Moreover, we have estimated the number of documented identifiable deaths to be 191,874±1772, which very closely matches the 2014 HRDAG estimate, completed by hand-matching. To our knowledge, we have the first estimate for the

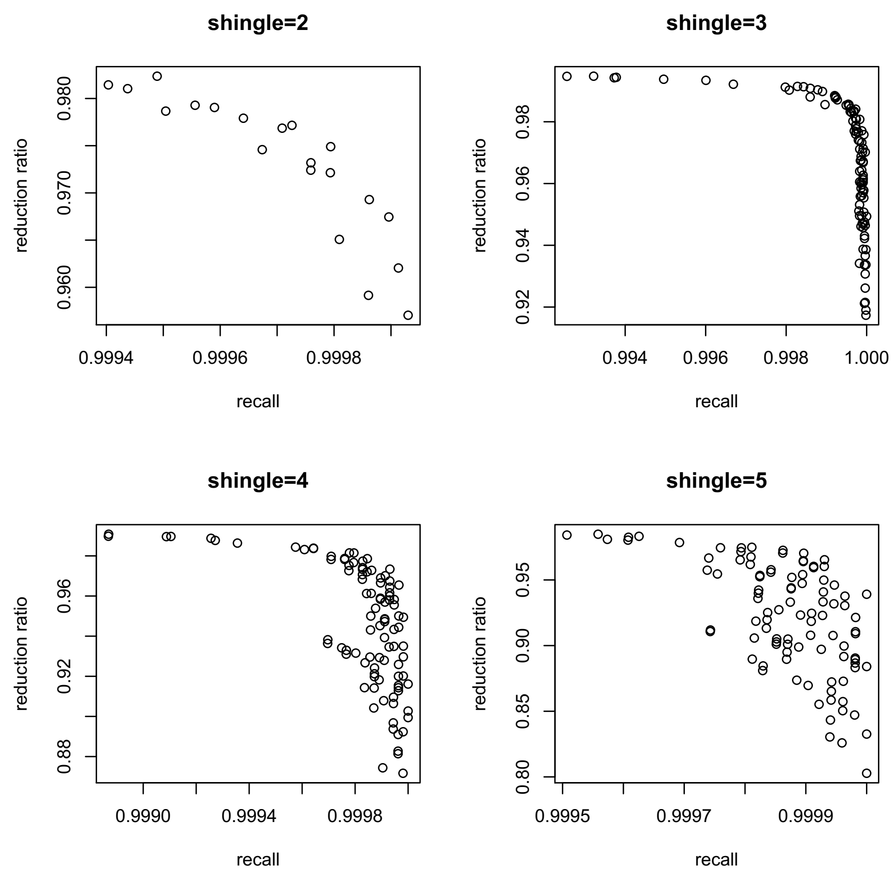

Fig 6: For shingles 2–5, we plot the RR versus the recall. Overall, we see the best behavior for a shingle of 3, where the RR and recall can be reached at 0.98 and 1, respectively. We allow L and K to vary on a grid here. L varies from 5–100 by steps of 5; and K takes values 15, 18, 20, 23, 25, 28, 30, 32, and 35.

number of documented identifiable deaths with a standard error associated with such an estimate. Our methods are scalable, potentially bringing impact to the human rights community, where such estimates could be updated in near real time. It could lead to further impact in public policy and transitional justice in Syria and other areas of conflict globally.

Acknowledgements: We would like to thank the Human Rights Data Analysis Group (HRDAG) and specifically, Megan Price, Patrick Ball, and Carmel Lee for commenting on our work and giving helpful suggestions that have improved the methodology and writing. We would also like to thank Stephen E. Fienberg and Lars Vilhuber for making this collaboration possible. PhD student Chen is supported by National Science Foundation (NSF) grant number NSF-1652131, NSF-1652431, NSF-1534412, and AFOSR-YIP FA9550-18-1-0152. Shrivastava's work is supported by NSF-1652131, NSF-1718478, AFOSR-YIP FA9550-18-1-0152, and an Amazon Research Award. Steorts's work is supported by NSF-1652431, NSF-1534412, and the Laboratory for Analytic Sciences (LAS). This work is representative of the author's alone and not of the funding organizations.

Supplementary Material

Supplementary Article: Supplementary Material for "Unique Entity Estimation with Application to the Syrian Conflict"
(doi: COMPLETED BY THE TYPESETTER; .pdf). This supplement consists of two parts. It offers more details about: (A) the Syrian data set and (B) our unique entity estimation proofs. in (A), we give details regarding the Syrian data set and the training data that is used. in (B), we give detailed proofs that our proposed estimator that is unbiased and has has provable low variance compared to random sampling. Refer to Chen, Shrivastava and Steorts (2018) for details.

References

Aleksandrov, P. S. (1956). *Combinatorial topology* **1**. Courier Corporation.

Andoni, A. and Indyk, P. (2004). E2lsh: Exact Euclidean Locality Sensitive Hashing Technical Report.

Baxter, R., Christen, P., Churches, T. et al. (2003). A comparison of fast blocking methods for record linkage. In *ACM SIGKDD* **3** 25–27.

Bhattacharya, I. and Getoor, L. (2006). A Latent Dirichlet Model for Unsupervised Entity Resolution. In *SDM* **5** 59. SIAM.

Broder, A. Z. (1997a). On the resemblance and containment of documents. In *Compression and Complexity of Sequences 1997. Proceedings* 21–29. IEEE.

Broder, A. Z. (1997b). On the Resemblance and Containment of Documents. In *the Compression and Complexity of Sequences* 21-29.

Chazelle, B., Rubinfeld, R. and Trevisan, L. (2005). Approximating the minimum spanning tree weight in sublinear time. *SIAM Journal on computing* **34** 1370–1379.

Chen, B., Shrivastava, A. and Steorts, R. C. (2018). Supplement to Unique Entity Estimate Applied to the Syrian Conflict. *Applied Annals of Statistics*.

Chen, B., Xu, Y. and Shrivastava, A. (2018). LSH Sampling breaks the Computational Chicken-and-Egg Loop in Adaptive Stochastic Gradient Estimation.

Christen, P. (2012). A survey of indexing techniques for scalable record linkage and deduplication. *IEEE Transactions on Knowledge and Data Engineering* **24** 1537–1555.

Christen, P. (2014). Preparation of a real voter data set for record linkage and duplicate detection research.

Deming, W. E. and Glasser, G. J. (1959). On the problem of matching lists by samples. *Journal of the American Statistical Association* **54** 403–415.

Erdos, P. and Rényi, A. (1960). On the evolution of random graphs. *Publ. Math. Inst. Hung. Acad. Sci* **5** 17–60.

Fellegi, I. and Sunter, A. (1969). A Theory for Record Linkage. *Journal of the American Statistical Association* **64** 1183–1210.

Frank, O. (1978). Estimation of the Number of Connected Components in a Graph by Using a Sampled Subgraph. *Scandinavian Journal of Statistics* **5** 177-188.

Gionis, A., Indyk, P., Motwani, R. et al. (1999). Similarity search in high dimensions via hashing. In *Very Large Data Bases (VLDB)* **99** 518–529.

Grillo, C. (2016). Judges in Habre Trial Cite HRDAG Analysis.

Gutman, R., Afendulis, C. C. and Zaslavsky, A. M. (2013). A Bayesian procedure for file linking to analyze end-of-life medical costs. *Journal of the American Statistical Association* **108** 34–47.

Indyk, P. and Motwani, R. (1998). Approximate Nearest Neighbors: Towards Removing the Curse of Dimensionality. In *STOC* 604-613.

Liang, H., Wang, Y., Christen, P. and Gayler, R. (2014). Noise-tolerant approximate blocking for dynamic real-time entity resolution. In *Pacific-Asia Conference on Knowledge Discovery and Data Mining* 449–460. Springer.

Liseo, B. and Tancredi, A. (2013). Some advances on Bayesian record linkage and inference for linked data. *URL http://www. ine. es/e/essnetdi_ws2011/ppts/Liseo_Tancredi. pdf*.

Luo, C. and Shrivastava, A. (2017). Arrays of (locality-sensitive) Count Estimators (ACE): High-Speed Anomaly Detection via Cache Lookups. *CoRR* **abs/1706.06664**.

Luo, C. and Shrivastava, A. (2018). Scaling-up Split-Merge MCMC with Locality Sensitive Sampling (LSS). *CoRR* **abs/1802.07444**.

McCallum, A., Nigam, K. and Ungar, L. H. (2000). Efficient clustering of high-dimensional data sets with application to reference matching. In *Proceedings of the sixth ACM SIGKDD international conference on Knowledge discovery and data mining* 169–178. ACM.

McCallum, A. and Wellner, B. (2004). Conditional Models of Identity Uncertainty with Application to Noun Coreference. In *Advances in Neural Information Processing Systems (NIPS '04)* 905–912. MIT Press.

Paulevé, L., Jégou, H. and Amsaleg, L. (2010). Locality sensitive hashing: A comparison of hash function types and querying mechanisms. *Pattern Recognition Letters* **31** 1348–1358.

Price, M., Klingner, J., Qtiesh, A. and Ball, P. (2014). Updated statistical analysis of documentation of killings in the Syrian Arab Republic. *United Nations Office of the UN High Commissioner for Human Rights.*

Provan, J. S. and Ball, M. O. (1983). The Complexity of Counting Cuts and of Computing the Probability that a Graph is Connected. *SIAM Journal on Computing* **12** 777-788.

Rajaraman, A. and Ullman, J. D. (2012). *Mining of massive datasets*. Cambridge University Press.

Sadinle, M. et al. (2014). Detecting duplicates in a homicide registry using a Bayesian partitioning approach. *The Annals of Applied Statistics* **8** 2404–2434.

Sadosky, P., Shrivastava, A., Price, M. and Steorts, R. C. (2015). Blocking Methods Applied to Casualty Records from the Syrian Conflict. *arXiv preprint arXiv:1510.07714*.

Shrivastava, A. and Li, P. (2014a). Densifying one permutation hashing via rotation for fast near neighbor search. In *Proceedings of The 31st International Conference on Machine Learning* 557–565.

Shrivastava, A. and Li, P. (2014b). Improved Densification of One Permutation Hashing. In *Proceedings of The 30th Conference on Uncertainty in Artificial Intelligence*.

Shrivastava, A. and Li, P. (2014c). In Defense of Minhash over Simhash. In *Proceedings of the Seventeenth International Conference on Artificial Intelligence and Statistics* 886–894.

Spring, R. and Shrivastava, A. (2017a). A New Unbiased and Efficient Class of LSH-Based Samplers and Estimators for Partition Function Computation in Log-Linear Models. *arXiv preprint arXiv:1703.05160*.

Spring, R. and Shrivastava, A. (2017b). Scalable and sustainable deep learning via randomized hashing. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* 445–454. ACM.

Steorts, R. C. (2015). Entity Resolution with Empirically Motivated Priors. *Bayesian Analysis* **10** 849–875.

Steorts, R. C., Hall, R. and Fienberg, S. E. (2014). SMERED: A Bayesian Approach to Graphical Record Linkage and De-duplication. *Journal of Machine Learning Research* **33** 922–930.

Steorts, R. C., Hall, R. and Fienberg, S. E. (2016). A Bayesian Approach to Graphical record Linkage and De-duplication. *Journal of the American Statistical Society*.

Steorts, R. C., Ventura, S. L., Sadinle, M. and Fienberg, S. E. (2014). A Comparison of Blocking Methods for Record Linkage. In *International Conference on Privacy in Statistical Databases* 253–268.

Tancredi, A. and Liseo, B. (2011). A hierarchical Bayesian approach to record linkage and population size problems. *Annals of Applied Statistics* **5**

1553–1585.

Vatsalan, D., Christen, P., O'Keefe, C. M. and Verykios, V. S. (2014). An evaluation framework for privacy-preserving record linkage. *Journal of Privacy and Confidentiality* **6** 3.

Wang, Y., Shrivastava, A. and Ryu, J. (2017). FLASH: Randomized Algorithms Accelerated over CPU-GPU for Ultra-High Dimensional Similarity Search. *ArXiv e-prints*.

Winkler, W. E. (2004). Approximate String Comparator Search Strategies for Very Large Administrative Lists. *Proceedings of the Section on Survey Research Methods, American Statistical Association.*

Winkler, W. E. (2006). Overview of record linkage and current research directions. In *Bureau of the Census*. Citeseer.

Zanella, G., Betancourt, B., Miller, J. W., Wallach, H., Zaidi, A. and Steorts, R. (2016). Flexible Models for Microclustering with Application to Entity Resolution. In *Advances in Neural Information Processing Systems* 1417–1425.