Generating Synthetic Data for Privacy Research

Zornitsa Asanska

4th Year Project Report
Computer Science and Management Science
School of Informatics
University of Edinburgh

2022

Abstract

Characterising people's behaviour over the Internet is of paramount importance for better understanding and enhancing their privacy. The availability and analysis of data depicting user online behaviour can assist in achieving this goal. However, the sensitive nature of the data makes its release difficult and in some cases illegal. In an attempt to address the lack of email communication data, we implement a statistical approach to generating statistically- and structurally-similar synthetic data of email traffic over a network. We further identify the need for hiding the degree distribution of the underlying communication network and implement a differentially private degree distribution approximation algorithm [19]. A set of statistical measures is used to evaluate the privacy-utility trade-off of approximating the original degree distribution. Finally, we present the simulation results of the real and synthetic email transactions over the anonymous communication system Loopix [35] and quantify the anonymity level of each simulation.

Research Ethics Approval

This project was planned in accordance with the Informatics Research Ethics policy. It did not involve any aspects that required approval from the Informatics Research Ethics committee.

Declaration

I declare that this thesis was composed by myself, that the work contained herein is my own except where explicitly stated otherwise in the text, and that this work has not been submitted for any other degree or professional qualification except as specified.

(*Zornitsa Asanska*)

Acknowledgements

I would like to begin this section by addressing my supervisor, Tariq Elahi. I wish to thank you for supporting my work and sharing your expertise throughout this project. I am beyond grateful for the academic as well as emotional advice I received during the rough patches I found myself in over the course of this year. Thank you for guiding my progress and taking interest in my ideas. I genuinely enjoyed our meetings which would often extend far beyond the scheduled time.

I would like to thank my two flatmates, who I can proudly say have become my Edinburgh family. To Saskia, thank you for the late nights in the library, for introducing a healthy sleep routine into my life and for keeping me sane in the weeks leading to the completion of this project. I would be happy if I had been even half as supportive of you as you were of me in a time which was equally hard for both of us.

I would also like to thank my friends, Ioana, Linda and Lucia, for taking the time to read through this report and provide me with their invaluable feedback. Thank you for opening my eyes to my mistakes, which I would not have otherwise discovered.

Finally, I would like to thank my family for their constant support not only throughout my work on this project but for the entirety of my studies. Thank you for your patience, love, encouragement and for always being ready for a late night call. Here, if I may, I wish to dedicate the next few words to them in my native language:

Мамо, тате, благодаря Ви за грижите, обичта и неизменната, нестихваща подкрепа през четирите години на моята бакалавърска степен. Благодаря Ви за проявеното търпение и разбиране през последните месеци, за късните разговори и ранните събуждания. Благословена съм от Бог точно Вие да сте моя опора и мой пример за подражание.

Contents

Chapter 1

Introduction

1.1 Motivation

Following technological advancements in data collection, processing and storage, more data is being collected in a wide variety of fields. The availability of large datasets creates an opportunity for further leveraging the data for academic and social purposes. In particular, progress in computer science and machine learning has been tightly linked to the availability of public datasets. Moreover, in many areas of science, new discoveries are the result of the research and analysis of relevant data [27]. A study of research papers in biology has shown higher citation rates and data reuse of research papers with publicly available data compared to similar studies without available data. The study further concludes a correlation between the increase in dataset availability and the increase in data analysis papers [36].

Research of user online behaviour is no different. The analysis and characterisation of user behaviour over a network provides a basis for the understanding and better protection of people's privacy across the Internet. Large data collections depicting users' behaviour (emails, text messages, website visits and navigation) are needed to achieve an accurate picture of people's behavioural patterns online. This poses a privacy paradox: privacy can be enhanced and better understood by first obtaining and studying private user information.

While the availability of data is paramount for technological advancement, the release and use of data threatens the privacy of the individuals involved. Due to the sensitive nature of the data, its publication and use becomes difficult and, in particular cases, even illegal under regulations such as GDPR in the European Union [30] and DPA in the United Kingdom [15].

A way to address the lack of data and circumvent the risks associated with its release is to use synthetic data which does not contain information related to any real individual. This motivates the primary goal of this project as the development of a synthetic data generation framework which models the email traffic within a network, while hiding the underlying social network to prevent the identification of accounts

based on the number of users they interact with.

1.2 Project Goals and Contributions

The initial proposal lays out the primary aim of the project as follows:

Create a synthetic data generation framework for emails.

The proposal further specifies the following completion criteria:

- *identify minimal meta-data items that provide good closeness to the real data*
- *evaluate the resultant privacy leakage of the generated datasets as compared to the real datasets*

The project description does not provide a concrete definition of privacy, leaving it to us to formalise this concept and evaluate the synthetic dataset with respect to the chosen definition. The project explores privacy in terms of the protection of individual identity in data generation and dataset release, as well as privacy in the context of anonymous communication over a network.

In response to the project goals, this dissertation focuses on modelling email traffic in a network. It implements a data generation framework which leverages a real dataset through statistical methods to generate statistically-similar synthetic set of email transactions, represented by a timestamp, origin and destination. Synthetic data is generated from a real email log dataset which captures the emails sent between a community of users within an academic setting in the School of Informatics at the University of Edinburgh.

To quantify privacy leakage, we simulate sending email messages over an anonymous MixNet communication network and compare the anonymity level of the simulation for the synthetic and original datasets.

The following key contributions have been made:

- Implementation of a Statistical Synthetic Email Data Generator for modelling email transactions for a predefined time period.
- Implementation of a graph projection algorithm for node ε-differentially-private degree distribution release.
- Transcript mode implementation of Loopix MixNet Simulator [35] to allow predefined email transactions to be simulated.
- Statistical evaluation of the synthetic data and evaluation of the importance of the graph structure for network communication privacy.

1.3 Report Outline

We begin this report by presenting related work and preliminary concepts in Chapter 2. The same chapter gives a summary of the two datasets which were used for the synthetic data generation. Chapter 3 summarises the mathematical definitions which will be referenced throughout the report. Chapters 4 and 5 hold the implementation details and main contributions of this project. Chapter 6 outlines the experiments and empirical evaluation of the synthetic data, along with the obtained results. Finally, in Chapter 7 we provide a final summary of the project. We critically assess both the implementation and evaluation processes described in this report and identify room for future work.

Chapter 2

Background

This chapter outlines preliminary concepts which form the starting point of this project. We begin by presenting related work, common algorithms for private data publishing and how they relate to the aims of this project. Section 2.1.3 presents the work of Babalola, et al. [2] – a set of statistical methods for stochastic synthesis of email transactions and main reference for the implementation of the Email Dataset Generator. Section 2.2 gives an overview of the differentially private algorithm for degree distribution approximation which has been implemented as part of the data generation process. A MixNet simulator is used to evaluate the anonymity level of the synthetic data in comparison to the original dataset. A summary of the anonymous MixNet communication system Loopix is presented in Section 2.3. Finally, Section 2.4 provides an overview of the two datasets which were used and the processing work carried out prior to data generation.

2.1 Data Release

Publishing sensitive personal data carries an inherent risk to people's privacy. Examples of failure to protect individual privacy in microdata release creates the societal need for data sharing with quantifiable privacy guarantees on an individual level [25], [28]. Two main classes of approaches have been introduced in an attempt to overcome this challenge.

2.1.1 Data Anonymisation and Sanitisation

The first class involves the application of algorithms which aim to remove identifying information from the dataset (de-identification). Such sanitisation and anonymisation approaches aim to produce a dataset which cannot be easily re-identified [38]. K-anonymity [39] provides a formal privacy definition to ensure that each individual is a member of a group of size at least k where all members share the same quasi-identifiers. This makes it possible to determine the membership of an individual to a group of at least k members and thus ensures the individual blends into the group.

While such formalisations exist, anonymisation algorithms often do not provide rig-

orous privacy guarantees and can be insufficient to protect individual privacy [11], [25], [28]. Methods have proven to be susceptible to linking and differencing attacks, which can re-identify individual records. Emam, et al. [10] presents the risks of re-identification of Canadian citizens based on demographic identifiers from a disclosed sanitised dataset. The publication of taxi trip records by the New York City Taxi and Limousine Commission (TLC) [40] has shown that the addresses and trip patterns of drivers can be extracted by de-anonymising car plate records in the data [31]. These examples prove that sanitisation does not provide sufficient protection of individual identity and establish the need for sharing personal data while protecting individual privacy.

2.1.2 Data Generation

The shortcoming of the first category of data privacy techniques motivates research into the second class which is based on the synthesis of data from original datasets so that released data does not contain any real information. Data generation methods aim to mimic the original data as closely as possible to ensure minimum decrease in data utility. Deep Learning models such as Generative Adversarial Networks (GANs) [14] and Variational autoencoders (VAE) [21] generate high-quality tabular synthetic data. Additionally, the statistical properties of data can be extracted and leveraged to generate synthetic datasets which are statistically and structurally similar to the original data [34]. Most data generation methods focus on modelling tabular data with single row record entries such as medical records. However, they do not extend to communication records of a social network and would fail to identify the characteristics of a social network graph and email sending patterns.

2.1.3 Social Network Communications

The goal of this project is to model email transactions in a network. The specific nature of the data requires the generation approach to be tailored to capturing the topology of the social network and the timings of email message correspondence. To synthesise the network communications over a network, the data can be characterised by answering the following two questions:

1. Which accounts communicate with each other?
2. When and how often do accounts communicate with each other?

Answering the first question relates to characterising the communication network as a graph structure and extrapolating its connectedness e.g. degree distribution, number of cliques of variable sizes. The social graph of an organisation can say a lot about its overall structure and hierarchy. This is true specifically for social networks which often exhibit a scale-free structure based on the degree distribution of their nodes [29]. Intuitively, scale-free networks - networks with degree distribution which follows a Power Law distribution, are characterised by a large number of nodes with a low degree and very few, highly-connected nodes (hubs). The small number and high connectivity of hub nodes makes them easily identifiable and poses a risk to the identity of the account holder. In an attempt to mitigate this risk, we have identified the importance

of hiding the structure of the social graph. This can be achieved through the means of differential privacy by adding calibrated random noise to the degree distribution of the graph. Subsection 2.1.4 provides an introduction into differential privacy and how it relates to the purposes of this project. Subsection 2.2 introduces the differentially private graph projection algorithm which is used in the generation process to hide the structure of the social network.

The second question relates to the temporal sending patterns of accounts within the social network. Mimicking those patterns can be achieved by extrapolating the email sending rates with respect to the time of the correspondence. Babalola, et al. [2] propose a statistical approach for generating email transactions by first extracting the weekly and hourly sending rate distributions and the individual activity of email accounts. Those are used as parameters to a time-dependent Poisson Process which is sampled to individually generate the email transactions of each node in the communication graph. We use the work of Babalola, et al. as a main reference for the implementation of the Email Dataset Generator.

2.1.4 Differential Privacy

On their own, generative methods do not ensure data privacy despite no actual 'real' data being released. Differential privacy (DP), a mathematical formalisation of privacy introduced by Dwork [8] allows privacy loss to be quantified and provides some of the strongest theoretical privacy guarantees. Rather than relating to the data itself, differential privacy places guarantees on the output of algorithms which query the data and release aggregate statistics. For an output of a differentially private algorithm on some dataset, the definition of differential privacy introduces an upper bound on the difference observed between the output of the algorithm when applied to that same dataset with a single record having been removed (neighbouring datasets). Intuitively, this means the two outputs would be indistinguishable relative to a privacy budget, thus revealing little information about the removed record. In this way, differential privacy provides protection against differencing attacks, which leverage aggregate statistics in an attempt to reveal information about an individual record by removing the single record and comparing the two outputs on the same query. The mathematical formalisation of these concepts has resulted in the development of differentially private data generation algorithms [44], [27], [43].

Differential privacy has been extended to graph data, where the difference lies in the definition of neighbouring datasets. An algorithm which satisfies graph DP guarantees ensures the removal of a single edge (edge-neighbouring graphs) or a single node along with all edges incident to it (node-neighbouring graphs) from the input graph does not change significantly the output of the algorithm [17]. Edge privacy corresponds to the guarantee that the properties of the relationship between two nodes (existence or absence) remain private. In our setting, an edge-DP algorithm will prevent conclusions about whether two accounts communicate. However, it would fail to protect the entire information of an individual in the network. In contrast, node-DP protects a node along with all of its adjacent edges. Data release under node-DP has more stringent privacy

guarantees compared to edge-DP but is more difficult to achieve without significant modification to the original data. We believe node-differential privacy is better suited for the goal of hiding the structure of the social network in a way which protects all relationships of an account. Therefore, we require an algorithm to privately release the degree distribution of the graph under node-DP.

Differentially private algorithms offer protection to individuals by enforcing a weak quantified correlation between the presence of a user (or node) in the data and the algorithm's output. In the context of the project, releasing the degree distribution and utilising it for the synthesis of a new graph under differential privacy ensures little correlation can be made between nodes from the original and the synthetic dataset based on their connectedness within the network.

Differential privacy is often achieved by adding calibrated noise to the algorithm output. Large amount of noise has to be added to the whole distribution, to hide nodes with large degree, resulting in heavy distortion and a decrease in the output's utility. This challenge has given rise to a key technique for satisfying node-differential privacy in graphs: *graph projection* - a category of transformation methods which project a graph onto a new θ-degree-bound graph. Kasiviswanathan et al. [19] propose a graph projection algorithm which transforms the graph by truncating nodes with large degree and adding calibrated noise to the resultant distribution. We choose this algorithm as it suites the privacy requirements outlined by the project's goals and provides a good trade-off between algorithm complexity and performance. Section 2.2 provides a summary of the algorithm.

All preliminary concepts and formal definitions of graph differential privacy are presented in Chapter 3.

2.2 Differentially Private Degree Distribution Approximation

Kasiviswanathan et al. [19] propose a graph projection algorithm which projects a graph into a new degree-bound graph. The algorithm truncates nodes with degree greater than a given threshold θ (naive truncation) and adds calibrated noise sampled from a Cauchy distribution to the truncated degree distribution.

The algorithm computes S_T, the smooth upper bound of the sensitivity of the truncation, i.e. the number of nodes whose degrees may change due to the removal of large-degree nodes. Instead of using the given degree bound θ, the cutoff threshold is randomised to obtain θ' and thus keep S_T low. The noise added to the truncated θ'-degree-bound distribution is calibrated based on the observation that changing a single node in a θ'-bounded graph can result in a change in up to $2\theta'$ nodes (when a node of degree θ' is rewired entirely by removing all its edges and creating the same number of edges none of which were previously present in the graph). This determines the scale parameter of the added Cauchy noise to be $\frac{2\theta'\sqrt{2}}{\varepsilon} S_T$.

The maximum value of the randomised upper bound θ' depends on the original value θ and on the number of nodes in the graph. For graphs with a large number of nodes (as is the case for the two datasets used in this project), this can result in an upper bound significantly higher than the original value of θ. To avoid such variation and explore the effect of truncating the graph at a predefined upper bound, we have decided not to randomise the degree upper bound and directly apply the truncation at θ.

2.3 Loopix Mixnet Simulator

Loopix is an anonymity communication system from the mix network family [35]. While the theoretical details behind its implementation are beyond the scope of the project, this section provides a summary of Loopix system and explains its relevance to the project.

Mix networks prevent sender-receiver communication tracing by introducing a set of proxy servers, also known as mixes or mix nodes, on the travel path of a message. Those mixes shuffle the order of messages following a predefined mixing strategy before sending them along (potentially to another mix node). Unlike classical mix networks, Loopix mixes don't wait until a fixed number of messages have been received before mixing and sending them. Instead, Loopix implements a simplification of the Stop-and-Go strategy which introduces a sending time window and a random delay sampled from an exponential distribution at each mix node [20]. It further leverages the Stop-and-Go strategy by introducing decoy cover traffic which follows a Poisson Process (Poisson Mixing). This provides a trade-off between anonymity, guaranteed by cover traffic and delays, and latency. The system is designed specifically for low-latency communication applications such as instant messaging and high-latency communication such as emails.

A Loopix simulator allows us to simulate the operation of the Loopix anonymity system and conduct experiments. The simulator provides measures which quantify message anonymity and end-to-end (E2E) unlinkability. It uses Shannon entropy [37] to measure the anonymity level of messages. Entropy is a measure of the uncertainty of a probabilistic event such as the identification of a sender from a set of potential senders (anonymity set). The higher the entropy, the less certainty an attacker has about the actual origin of the message (that is the attacker considers all of the potential senders in the anonymity set as equally likely to have sent the message) [7]. This allows us to obtain a measure of how distinguishable the sender is and thus determine the anonymity of messages sent over the system.

Additionally, the simulator measures the end-to-end (E2E) unlinkability of the communication as the *expected difference in the likelihood* that a message is sent from one sender in comparison to another:

$$\varepsilon = |log(p_0/p_1)|$$

where p_0 and p_1 are the probabilities that a message was sent by either one of two distinct potential senders.

Those metrics will be used to compare the privacy leakage of the original and synthetic datasets in the context of communication privacy. This will allow us to investigate the impact of the network graph structure on communication security. In this setting the goal of the comparison is not to achieve a high anonymity measure for the synthetic dataset but rather to evaluate the similarity of the original and synthetic datasets on the results of the simulation.

The current implementation of the simulator only supports sending messages from one client to another at random times based on a predefined sending rate. To allow sending messages at times specified by an email log, we have implemented a transcript mode, which extracts the email messages of each account from a *csv* file and generates delays to match the exact times a message is sent. The details of the implementation are presented in Chapter 5.

It is important to note that communication privacy relates to secure communications over a network, whereas differential privacy, as mentioned in the previous section, relates to individual identity protection in the context of data release.

2.4 Datasets

This subsection provides a description of the two reference datasets used in the data generation process and the data processing which was required prior to the data generation.

2.4.1 Enron Email Corpus

The Enron email corpus was made public by the Federal Energy Regulatory Commission in 2004 [4]. The database consists of 619,446 emails, grouped in folders, belonging to 158 employees of the senior management circle of the company. The emails were generated in the years 1999 to 2002 leading up to the company's collapse in 2001. Due to events surrounding the data release, research of the corpus has been primarily focused on analysis of the semantic content for fraud detection [33], email classification [22] [26] and network characterisation [5], [16].

It is important to emphasize that the Enron dataset was used as a reference when making decisions concerning the implementation of the data generation algorithm. The advantages and potential flaws of this approach have been discussed in Section 7.2.

2.4.1.1 Format

The database consists of email logs which contain the metadata and semantic content of messages, presented as a single string sequence of key-value pairs. Below is a single entry:

```
Message-ID: <30965995.1075863688265.JavaMail.evans@thyme>
Date: Thu, 31 Aug 2000 04:17:00 -0700 (PDT)
From: phillip.allen@enron.com
To: greg.piper@enron.com
Subject: Re: Hello
Mime-Version: 1.0
Content-Type: text/plain; charset=us-ascii
Content-Transfer-Encoding: 7bit
X-From: Phillip K Allen
X-To: Greg Piper
X-cc:
X-bcc:
X-Folder: \Phillip_Allen_Dec2000\Notes Folders\'sent mail
X-Origin: Allen-P
X-FileName: pallen.nsf

Greg,
 How about either next Tuesday or Thursday?
Phillip
```

2.4.1.2 Processing

The database consists of information irrelevant for the purposes of the project: invalid entries, missing fields and email duplicates. Considerable effort was put into cleaning the dataset.

- **Extraction**
  The project is not concerned with the textual content but rather with the origin, destination and timestamp of the email transactions. Therefore, only the relevant set of fields is extracted from each entry ('Message-ID'; 'Date'; From: ['From', 'X-From']; To: ['To', 'X-To', 'X-cc', 'X-bcc']).

- **Duplicate removal**
  Most of the duplicate emails reside in folders such as 'discussion_threads', 'all_documents', 'sent_mail', which seem to be computer-generated, and all of their contents are already present in other user-created folders or in the main mailbox of the user [45]. The dataset contained multiple entries with the same Message-Id, usually due to a copy being stored by both the sender and recipient.

- **Invalid Entries**
  Entries with invalid or missing addresser or addressee were discovered using regular expressions. The majority of messages in the dataset were sent between 1999 and 2002 but some entries included a timestamp dating back to 1979 or on the other end – stating dates in the future such as the year 2040.

- **Sanitisation**
  The Enron database contains the original email addresses of the correspondents, which had to be sanitised. A simple sanitisation technique was used to anonymise

the data by generating a mapping for all email addresses to a randomly generated unique id.

All duplicate and invalid messages described above were removed.

2.4.2 School of Informatics: Sendmail logs

The Sendmail logs were obtained from the School of Informatics at the University of Edinburgh and include email communications of the IT Services at the university over the course of 2 months between 31 Nov 2019 and 31 Jan 2020. The logs have been sanitised prior to our receipt of the dataset.

2.4.2.1 Format

Each line in the system log consists of a timestamp, the name of the machine that generated the log (`host`), the word `sendmail:`, Process-id (`pid`), Queue-id (`qid`) and a message. Most messages are a sequence of `name=value` pairs [1]:

```
<date> <host> sendmail[pid]: <qid>: [<name>=<value>,]
```

Two log entries corresponding to the same email message are presented below with some `name=value` pairs omitted.

```
2020-01-31T15:32:03.970293+00:00 crunchie sendmail[8691]:
00VFW3CK008691:
from=<ead680ca1f3344f2fa4b9dc297bcd7>,
size=26196,
class=-60,
nrcpts=1,
msgid=d4017bcdd7801840374de261a2e724f,

2020-01-31T15:32:04.536097+00:00 crunchie sendmail[8699]:
00VFW3CK008691:
to=<76e99001bc50dc6bbb7cd8312506d1>,
delay=00:00:01,
xdelay=00:00:01,
mailer=esmtp,
pri=164385,
dsn=2.0.0,
stat=Sent (00VFW4xK013490 Message accepted for delivery)
```

2.4.2.2 Processing

Each log of the database was parsed using regular expressions to extract the following fields: `<date>`, `<qid>`, `<from_value>`, `<to_value>`, `<msgid_value>`, `<stat_value>`. As in the example, the database contains multiple log entries for a single email message. Logs corresponding to the same message share the same queue-id, which made it easy to group the logs and extract the time, origin and destination of each message.

Any messages with invalid status value (`stat`), such as `User unknown`, `Deferred` or `Data format error` were dropped.

2.4.3 Summary

The processed datasets are stored as *csv* files, where every line corresponds to a message record, characterised by a timestamp, the id of the originating node (sender) and a set of destination ids (recipient list). Table 2.1 gives a summary of the underlying social graph of both datasets and Table 2.2 presents the total and weekly average number of messages for the entire datasets and the average and maximum number of messages sent by a single account. The average hourly email sending rates over the course of a week are shown on Figure 2.1.

	$\|V\|$	$\|E\|$	d_{avg}	d_{max}	Clustering Coefficient	Power Law Exponent
Enron	6096	76271	9.023	908	0.314	2.436
Sendmail log	3848	7400	4.920	1351	0.288	2.230

Table 2.1: Dataset information: Graph summary, where V is the set of nodes (email accounts), E is the edge set and d is a node's degree

	Start date	End date	$\|M\|$	$\|M\|_{weekly,avg}$	$\|M_i\|_{avg}$	$\|M_i\|_{max}$
Enron	01-01-2001	01-01-2002	156954	3018.4	25.74	4759
Sendmail log	30-11-2019	31-01-2001	87412	9712.4	29.08	9684

Table 2.2: Dataset information: Message summary, where M is the set of email messages and M_i is the set of messages originating from account i

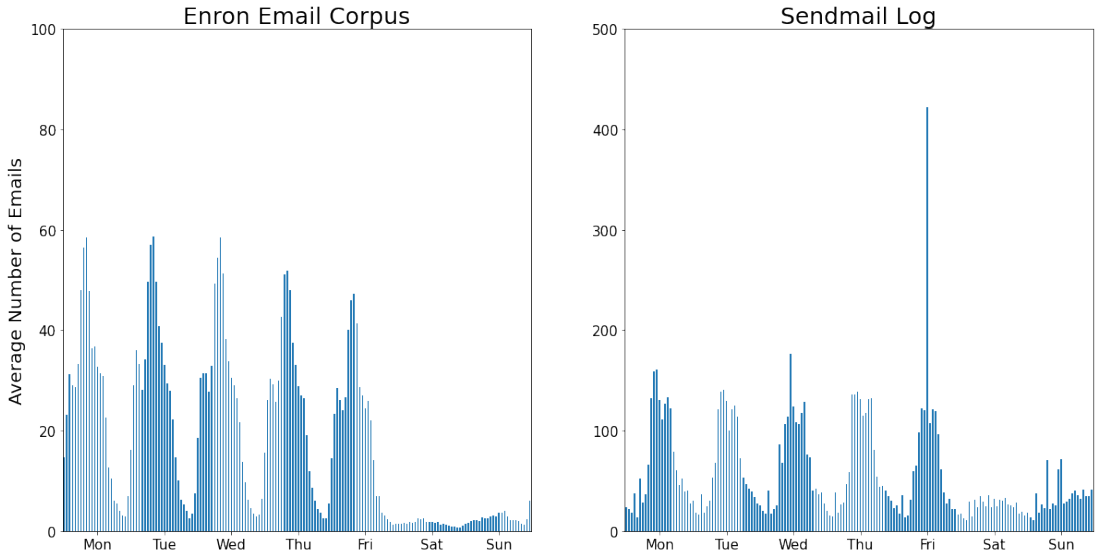


Figure 2.1: Average hourly email sending rates in a week

Chapter 3

Definitions

This chapter provides a formalisation of definitions and related concepts. The first section defines differential privacy and its variations relating to graph privacy: node- and edge-differential privacy. The second section is a mathematical formalisation of the evaluation metrics used in the project. Those concepts will be referenced throughout the report.

3.1 Differential Privacy

Definition 1 (**ε-Differential Privacy** [8])**.** A randomised algorithm A is ε-differentially private if for all two neighbouring datasets D, D' and any subset of outputs S in the output space of A: $S \subseteq Range(A)$, the following holds:

$$Pr[A(D) \in S] \leq e^{\varepsilon} \times Pr[A(D') \in S]$$

where Pr is the probability over the randomness of A, ε is the *privacy parameter*, also called privacy budget. Two datasets are neighbours if they differ by exactly one entry (whether modified or removed).

Intuitively, for some query on some dataset, a differentially private algorithm produces an output, regulated by the *privacy parameter* ε, that is statistically indistinguishable within a factor of e^{ε} from the same query on the same dataset had any one individual's information been excluded or added. That is, the exclusion or addition of a single sample of the dataset results in an insignificant change of the output of A, bounded by the privacy parameter. Smaller ε values enforce stronger privacy and require more noise to be added to ensure differential privacy, often at the expense of data accuracy and utility.

This definition of differential privacy refers to tabular datasets where a single entry represents the record of a single individual. In this case individual entries can be added, removed or modified without affecting the rest of the dataset. However, this definition has to be adapted when applied to a graphs. Hay et al. [17] propose two definitions: node (node-DP) and edge differential privacy (edge-DP).

Definition 2 (**ε-Node (Edge) Differential Privacy** [17])**.** A randomised algorithm A is ε-node private (resp. edge-private) if for all two graphs G, G' at node-distance 1 (resp. edge-distance 1) and any subset of outputs S in the output space of A: $S \subseteq Range(A)$, the following holds:

$$Pr[A(G) \in S] \leq e^{\varepsilon} \times Pr[A(G') \in S]$$

where the two datasets are neighbours if they differ by exactly one entry.

The difference between node-DP and edge-DP depends entirely on the definition of neighbouring graphs. Neighbouring graphs in the context of edge-DP differ by a single edge which has been removed from the dataset. The definition of Node Neighbouring graphs is presented below.

Definition 3 (**Node Distance**)**.** For two graphs G and G', the node distance $d_{node}(G, G')$ is defined as the minimum number of nodes which need to be modified in G to obtain G'.

$$d_{node}(G, G') = max(|V_G|, |V_{G'}|) - k$$

where k is the number of nodes in the largest induced subgraph of G which is equal to the corresponding induced subgraph of G'. Node modification includes the insertion or removal of the node and its set of edges, or a modification to the adjacency list of the node.

Definition 4 (**Node Neighbours**)**.** Two graphs G and G' are node neighbours if $d_{node}(G, G') = 1$.

Differential privacy of a randomised algorithm f is achieved by adding random noise where the amount of added noise is calibrated to the sensitivity of the algorithm [9]. Intuitively, the sensitivity of a function is the quantified difference which can be observed in the output of f when applied to two neighbouring graphs G and G'.

Definition 5 (**Local Sensitivity**[9])**.** For a function $f : \mathcal{G}_n \to \mathbb{R}^p$ and a graph $G \in \mathcal{G}_n$ with n nodes, the local sensitivity of f at G is:

$$LS_f(G) = max_{G'} ||f(G) - fG'||_1$$

where the maximum is taken from all node neighbours of G.
The higher the sensitivity, the more noise has to be added to mask the difference in the two outputs of the algorithm.

Edge-DP protects the relationship between any two nodes, whereas node-DP provides privacy to the node, including its existence in the dataset and its relations to other nodes [24]. The removal of a single node can subsequently lead to the removal of up to $|V|$ edges (where V is the set of nodes in the graph). Therefore, node-DP has higher sensitivity, especially in well-connected graphs. As such it provides stronger privacy guarantees but is harder to achieve [6].
Nodes with large degrees represent the main challenge in designing differentially private graph algorithms because they result in high sensitivity and thus require large amounts of noise to be added. To overcome this challenge, node-DP algorithms use graph projection to transform the original graph into a graph where nodes with large degrees are removed in an attempt to limit the sensitivity and consequently the amount of calibrated noise. Naive truncation is one such graph projection algorithm [19].

Definition 6 (**Naive Truncation**)**.** For a graph $G = (V, E)$ and an integer value θ, naive truncation is a function $T_\theta : \mathcal{G} \to \mathcal{G}$ defined as $T(V, E) = (V', E')$ where $V' = \{v | v \in V; d_v \leq \theta\}$ and $E' = \{(v_1, v_2) | (v_1, v_2) \in E; v_1, v_2 \in V'\}$.
Naive truncation is the operator which removes all nodes with degree greater than an upper bound θ along with all edges adjacent to them.

Definition 7 (**Smooth Bound on Local Sensitivity** [19])**.** For $\beta > 0$, a function $S : \mathcal{G}_n \to \mathbb{R}$ is an β-smooth upper bound on the local sensitivity of f if the following is satisfied:

$$\begin{aligned} \forall G \in \mathcal{G}_n : \quad & S(G) \geq LS_f(G) \\ \forall \, neighbours \, G, G' \in \mathcal{G}_n : \quad & S(G) \leq e^\beta S(G') \end{aligned}$$

Theorem 1 (**Smooth Bound on Composed Functions** [19])**.** *Let $T : \mathcal{G}_n \to \mathcal{G}_{n,\theta}$. If $S_T(G)$ is the β-smooth upper bound on the local sensitivity of T, then $S_{f \circ T}(G) = S_T(G) \times \Delta_\theta f$ is a β-smooth upper bound on the local sensitivity of $f \circ T$.*

Theorem 2 (**Calibrating Cauchy Noise to Local Sensitivity Smooth Bounds** [19])**.** *For a function $f : \mathcal{G}_n \to \mathbb{R}^p$, let S be its $\beta-$smooth bound on LS_f. For $\beta \leq \frac{\varepsilon}{\sqrt{2}p}$ an algorithm:*

$$A(G) = f(G) + Cauchy(\frac{\sqrt{2}}{\varepsilon} \times S(G))^p$$

is ε-differentially private.

Definition 8 (**Smooth Bound on Naive Truncation**)**.** The smooth upper bound on the local sensitivity of naive truncation $S_{T_\theta} : \mathcal{G} \to \mathcal{G}$ is defined as:

$$S_{T_\theta}(G) = max_k e^{-\beta k}(1 + k + N_k(G))$$

where $k \in [1, \ldots, \theta]$ and $N_k(K)$ is the number of nodes in G with degrees in the range $[\theta - k, \theta + k + 1]$. Then $(1 + k + N_k(G))$ corresponds to the local sensitivity of truncation for any two graphs G and G' with node distance k.

Intuitively, the smooth upper bound on naive truncation is a bound on the number of nodes in the graph which can experience a change in their degree as a result of the truncation. Based on Theorem 1 and 2 and Definition 9, Kasiviswanathan et al. [19] obtain a truncation algorithm which adds calibrated noise to the discrete degree distribution of θ-degree-bound graph G:

Theorem 3 (ε-differentially Private Degree Distribution Approximation)**.** *Let $p \in \mathbb{R}^{d_{max}+1}$ be the degree distribution of graph G_n with maximum node degree d_{max} and $p_{T_\theta} \in \mathbb{R}^{\theta+1}$ be the degree distribution of the graph after truncation. Then the degree distribution $p_{T_{\theta,\varepsilon}}$:*

$$p_{T_{\theta,\varepsilon}} = p_{T_\theta} + Cauchy(\frac{\sqrt{2}}{\varepsilon} \times 2\theta \times S_{T_\theta}(G))^{\theta+1}$$

is ε-differentially private.

3.2 Evaluation Metrics

As most differentially private algorithms, the truncation algorithm relies on adding noise to the degree distribution of the original graph, which can come at the cost of data utility. This section presents the metrics used to quantify the utility loss of hiding the social graph structure.

Definition 1 (PER: Preserved Edge Ratio). For an original graph $G(V,E)$ and a projected graph $G^{\theta}(V^{\theta},E^{\theta})$, the preserved edge ratio is defined as the ratio of the number of edges in the θ-bounded graph G^{θ} and the number of edges of the original graph G.

$$PER = \frac{|E^{\theta}|}{|E|}$$

While PER does not give any insight into the way edges are distributed within the graph, it provides a measure of the overall number of edges in the synthetic graph in comparison to the original graph of the same size. An optimal value would be $PER = 1$.

Definition 2 (L1 Distance). The L1 norm of two distributions $d_{\mathcal{G}} \in \mathbb{R}^m$ and $d_{\mathcal{G}'}\mathbb{R}^{m'}$:

$$||d_{\mathcal{G}} - d_{\mathcal{G}'}||_1 = \sum_{i=1}^{n} |d_{\mathcal{G},i} - d_{\mathcal{G}',i}|$$

where, in the context of the project, $d_{\mathcal{G}}$ and $d_{\mathcal{G}'}$ are the discrete degree distributions of graphs G and G', respectively. $d_{\mathcal{G},i}$ is the number of edges with degree i in $\mathcal{G}$. The distribution with the lower dimension is padded with zeros for the comparison and $n = max(m,m')$.

Definition 3 (Kolmogorov-Smirnov (KS) Distance). The KS distance of the cumulative distribution functions of two distributions is used to measure how close the two distributions are:

$$KS(d_1,d_2) = max_i|CDF_{d_1}(i) - CDF_{d_2}(i)|$$

Definition 4 (Graph Average Clustering Coefficient [42]**).** The Average clustering coefficient of graph $G = (V,E)$ is defined as the average of the local clustering coefficients of all vertices in V. The local clustering coefficient is a measure of how well connected the neighbours of a node are within each other.
Formally, the local clustering coefficient of node v is the number of edges between the neighbours of v and the maximum number of edges that could exist between them:

$$C = \frac{1}{|V|}\sum_{v_k \in V} \frac{2 \times |\{e_{i,j}|v_i, v_j \in N_k, e_{i,j} \in E\}|}{|N_k| \times (|N_k| - 1)|}$$

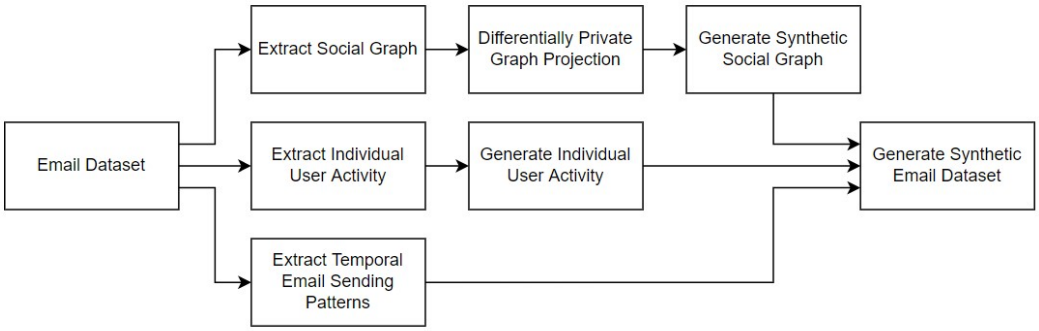
where N_k is the set of vertices directly adjacent to vertex v_k.

Chapter 4

Implementation: Email Data Generator

This chapter presents the design choices and implementation of the Email Data Generator. The Email Data Generator provides an implementation of a statistic data generation approach suited for email log datasets. It infers the structure of the communication network (which nodes communicate with each other), and the temporal patterns of these communications (when do nodes communicate with each other). Those data characteristics are used to stochastically generate synthetic email traffic.

4.1 System Overview



The data extraction process consists of extracting statistical measures to characterise:

- the social graph constructed by email transactions between accounts;
- the activity level of each account determined by the number of emails they have sent;
- the temporal patterns of email traffic.

Those are subsequently sampled to generate synthetic email messages.

4.2 Network Graph

The network of the dataset is represented as an undirected graph where each email account is a vertex and an email message from one user to another is an edge between two nodes. The degree of each node is extracted from the network to form a degree distribution which describes the connectedness of the graph.

4.2.1 Fitting the Original Degree Distribution

When plotting the degree distribution histogram (Figure 4.1), we see a large number of nodes with low degree and a very small number of highly connected nodes (hubs). In that sense the degree distribution resembles that of a *scale-free* network. A scale-free network is one with a Power Law degree distribution, where the proportion of nodes with degree d is directly proportional to $d^{-\alpha}$:

$$Pr_{deg}(d) \propto d^{-\alpha}$$

We used the `powerlaw` library in `Python` to obtain an optimal fit of the degree distribution. The result is an α parameter equal to 2.436.

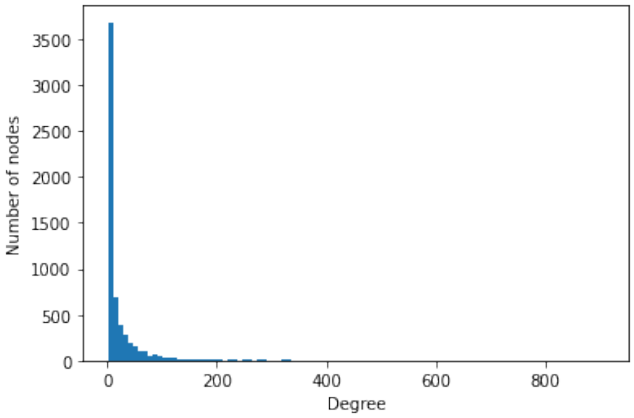


Figure 4.1: Degree Distribution Histogram

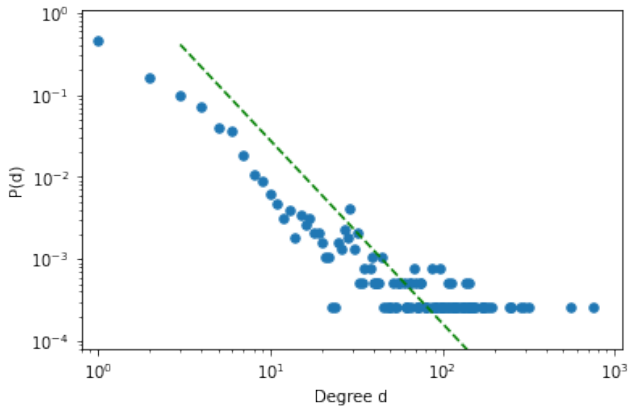


Figure 4.2: Log-log Degree distribution

4.2.2 Node-DP Degree Distribution Approximation

To hide the original structure of the graph, the Email Data Generator incorporates a variation of the truncation algorithm proposed by Kasiviswanathan et al. [19] for node-differentially private degree distribution approximation. The algorithm outputs a private θ-bound degree distribution. It randomises the actual upper bound θ′ used in the truncation by randomly selecting it from the range:

$$\theta' \backsim \{\theta + (\frac{\sqrt{2}(\theta+1) \times ln|V|}{\varepsilon}) + 1, \ldots, 2\theta + (\frac{\sqrt{2}(\theta+1) \times ln|V|}{\varepsilon})\}$$

The greater the initial value θ and the number of nodes in the graph $|V|$, the higher the upper bound of the randomised value θ′ becomes. As a result, graphs with many nodes will experience a degree upper bound θ′ which is significantly higher than the initial value of θ. For that reason, we have decided not to randomise the upper bound and directly perform truncation with upper bound θ. The pseudocode of the implementation is presented as Algorithm 1.

Algorithm 1 ε-Node-DP Algorithm for Degree Distribution Approximation

Input: ε, θ, $G = (V, E)$, p - frequency degree distribution of G

1: $b \leftarrow \varepsilon/\sqrt{2}(\theta+1)$
2: $S_{max} \leftarrow 0$
3: **for** $k = 1, 2, \ldots, \theta$ **do**
4: $\quad n \leftarrow |\{v | v \in V, deg_v \in [\theta - k, \ldots, \theta + k + 1]\}|$ ▷ Number of nodes with degree in range
5: $\quad S_{max} \leftarrow max(S_{max}, e^{-bk}(1 + k + n))$ ▷ Smooth upper bound of LS of Naive Truncation
6: **end for**
7: $c \backsim Cauchy(\frac{\sqrt{2}}{\varepsilon} \times 2\theta \times S(G))^{\theta+1}$ ▷ Random noise
8: $p' \leftarrow p_{deg \leq \theta} + c$
9: **Output:** p'

Steps 3-5 determine the smooth upper bound of the local sensitivity of the graph truncation and steps 7,8 generate the node-private degree distribution by adding random noise sampled from Cauchy distribution to the θ-bound degree distribution.

Unlike the degree distribution of the original social graph, the node-DP degree distribution cannot be modelled using a Power Law distribution due to the truncation of large degrees. Instead, the random noise is added to the node frequency, which is then normalised to obtain a discrete distribution. The private distribution is sampled when generating the degrees of nodes in the synthetic graph.

4.2.3 Graph Generation from Prescribed Degree Sequence

The node degree distribution (whether discrete or Power law) is sampled to generate the degrees of the nodes for the synthetic network graph. The connectivity graph

is generated following the sequential algorithm for generating random graphs with prescribed degrees, presented by Blitzstein and Diaconis [3]. The algorithm generates the edges of a graph by iteratively populating a binary matrix. The pseudocode in Algorithm 2 describes the algorithm:

Algorithm 2 Random Graph Generation with Prescribed Degrees

Input: Degree Sequence of size N: $D = (d_1, d_2, \ldots, d_N)$ in decreasing order
1: **if** $D.sum$ is odd **then**
2: $D_N \leftarrow D_N + 1$
3: **end if**
4: $R \leftarrow []$ ▷ Residual Degrees
5: **for** $i = 1, 2, \ldots, N$ **do**
6: $R \leftarrow R + [i] * D_i$
7: **end for**
8: $A \leftarrow N \times N$ 0-matrix ▷ Adjacency Matrix
9: $m \leftarrow 100$ ▷ Max Retry Constant
10: $c \leftarrow 0$ ▷ Retry Count
11: **while** R not empty **do**
12: $i \leftarrow R_1$
13: $f \leftarrow$ random from $\{2, 3, \ldots, R.size\}$
14: $j \leftarrow R_f$
15: **if** $i = j$ or $A_{i,j} > 0$ **then**
16: $c \leftarrow c + 1$
17: **if** $c >= m$ **then** ▷ Max retry reached; Restart Algorithm
18: Return to Step 8
19: **end if**
20: Return to Step 14
21: **end if**
22: $A_{i,j}, A_{j,i} \leftarrow 1$
23: Del R_1, R_f
24: **end while**
25: **Output:** A

4.3 Individual User Activity

The generation method is based on the assumption that some individual users are more active and send more emails compared to others. Applying k-means clustering on the total number of emails sent by each individual user, allowed us to identify different activity clusters. The elbow rule was used to determine the most appropriate number of activity clusters in the dataset (Figure 4.3). This number is entirely dataset specific and will differ for different original datasets.

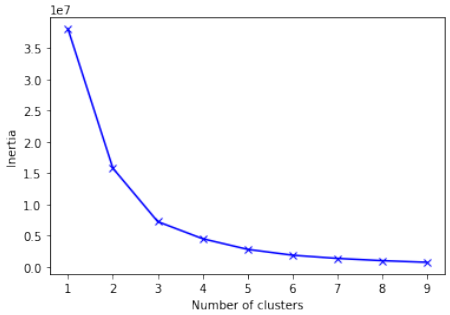


Figure 4.3: Number of activity clusters against distortion for Enron dataset

A discrete distribution is used to capture the allocation of nodes across the different activity clusters, where each cluster can be thought of as a different activity level. When implementing this for the Enron email dataset, most nodes end up in a cluster of relatively inactive accounts with a small number of sent emails (appearing as the blue cluster in Figure 4.4).

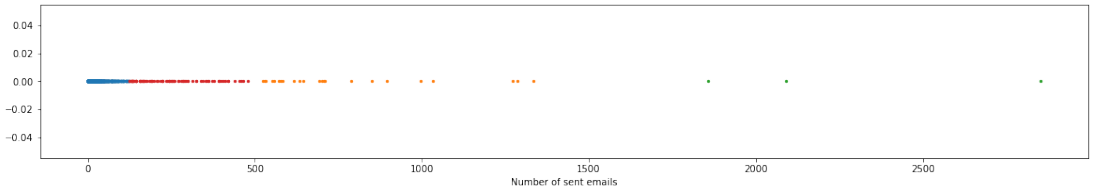


Figure 4.4: Account distribution in 4 Activity Clusters

A separate activity distribution is extracted for the nodes in each cluster. The distributions represent the total number of emails sent by each node.

The activity cluster distribution is sampled N times to generate the activity level of the N synthetic nodes. Depending on the activity level of each node, the corresponding activity distribution is sampled to determine the exact activity proportion of each node. The final result is a $\delta \in \mathbb{R}^N$ normalised vector of activity proportions for the N individuals in the synthetic network. This vector is used to normalise the total sending rate $\lambda(t)$ once emails are generated for each node individually.

4.4 Capturing Temporal Patterns of Email Traffic

The temporal patterns of sending emails across the social network are characterised by the timestamp of the messages. Those patterns depend on the nature and origin of the dataset. The Enron dataset contains email messages sent over the course of a whole year, which plays a big role in determining how to extract email sending rates. Given the Enron dataset holds email communications from a work environment, there seem to be a large number of emails sent during working hours on business days and a clear decrease in communications outside of working hours and during the weekend, which motivates our choice of timestep to be one hour in a weekly model. A smaller

timestep might be more appropriate if there are more fine-grained patterns in the data. For the Enron dataset, the emails generated within an hour seem to follow a uniform distribution so emails a equally likely to be generated at any minute within the course of an hour. In addition, the average number of emails sent in an hour is 10.69, therefore, the sending rates would become really small should a smaller timestep (e.g. minute) be chosen.

As a result, the timestamps of all email messages are used to obtain a time-dependent function λ_t of the sending rates of emails. The function outputs the expected number of generated emails across all accounts given a specific hour of the week.

4.5 Email Message Generation

The generation of individual emails follows a time-dependent arrival function which can be modelled using an Inhomogeneous Poisson Point process. The rates extracted at the previous stage are now used as email-generation rates at time-specific intervals aggregated across all accounts. The rate is then normalised using the individual activity of each account to get the email generation rate for an account at a specific hour of the week.

4.5.1 Time-dependent Poisson Process Simulation

We simulate a homogeneous Poisson Point Process in Python by setting a constant rate r to draw N samples. However, email sending rates vary depending on the time of generation. This is formally know as a Nonhomogeneous Poisson Process, when the intensity of a Poisson Process is not a constant but a location-dependent function in the underlying space on which the Poisson process is defined [12].

For the generation of a Nonhomogeneous Poisson Process, thinning has been applied, also known as acceptance-rejection method. The intuition behind the thinning algorithm is to find an upper-bound rate λ_{max} which dominates the rate function $\lambda(t)$. The rate λ_{max} is used to simulate a Poisson Process with a constant rate. Then an appropriate fraction of the generated the events is rejected until the rate $\lambda(t)$ is achieved. The details and proof of correctness of the the method are presented by R.Pasupathy [32].
The rate parameter λ is a function of time, which can be interpreted as an hourly email sending rate depending on the time of day and week. The function is deterministic and has a finite input space limited by the weekly-hourly model, which simplifies the simulation process. The following procedure illustrates the steps involved in simulating the upper-bound Homogeneous Poisson Process and thinning tailored to the email generation.

Algorithm 3 Poisson Process Simulation

Input: Individual activity proportion δ, Time-dependent rate function λ, Time t, Upper-bound rate λ_{max}

1: Generate $p \backsim Poisson(\lambda_{max} \times \delta)$ ▷ Number of potential emails
2: $r \leftarrow 0$ ▷ Number of realised emails
3: **for** $i = 1, 2, \ldots, p$ **do**
4: Generate $g \backsim U(0,1)$ ▷ Draw from Normal distribution
5: **if** $g < {}^{\lambda(t)}/_{\lambda_{max}}$ **then**
6: $r \leftarrow r + 1$ ▷ Realised Email
7: **end if**
8: **end for**
9: **Output:** r

Steps 5-7 are equivalent to drawing a sample from a Bernoulli distribution with parameter ${}^{\lambda(t)}/_{\lambda_{max}}$, which either accepts or rejects the potential email.
The algorithm is run for every node in the synthetic graph and every time interval within the generation period to determine the number of emails sent at each time interval. Email generation rates seem to follow a uniform distribution within the chosen time interval of 60 minutes. This means an email is equally likely to be generated at any minute within an hour. Consequently, to generate the exact times of emails messages, the time is chosen randomly within the one-hour time interval.

4.5.2 Recipient Selection

Each email account has a set of the exact times at which an email message is sent out. We use the binary adjacency matrix of the synthetic graph to select a list of recipients for each message. The recipients of a message are a randomly chosen subset of the sender node's neighbours.

Chapter 5

Implementation: Transcripted Loopix MixNet Simulation

The Loopix MixNet Simulator is used for the empirical evaluation of the Loopix communication system. It consists of a test mode which simulates the generation of 'real' messages at a predefined rate. For the purposes of this project we wish to evaluate the synthetic dataset of email communications against the original dataset in the context of network security. This required the implementation of a transcript mode which only sends messages at specific times determined by a log file. The chapter begins with a summary of the existing system and continues to outline the details of the transcript mode implementation.

5.1 System Overview

The simulation is a multi-agent system, where each node in the network, whether a mix or a client (sender or recipient), is implemented as an individual process. The implementation is in Python using the `SimPy` package for discrete-event simulation. It leverages OOP principles by introducing a parent `Node` class which is responsible for packet handling. It implements the sending, mixing, forwarding and receipt of real and cover traffic. The `Client` class represents the senders and recipients in the network, and holds the main logic of the transcript mode. All clients and mixnodes are initialised as part of a `Network` class instance which initialises the network layers and the mix nodes in each layer. Communications between nodes are represented by `Message` instances with randomly generated payload split into `Packets`, which are transported individually over the network.

A configuration `json` file provides the simulation settings - network topology, number of clients, sending rate and delays, message and packet size. The current design of the system was taken into account to minimise the number of changes when implementing the transcript mode.

5.2 Delay Generation

The simulator creates a separate process and simulates messages individually for each client. It uses delays to determine the amount of time in-between messages and schedules timeouts before sending each new message.

The log file contains the time, origin and destination of each message ordered by timestamp for all email accounts. That meant the log file had to be parsed by extracting the messages of each client to allow each `Client` instance to individually schedule messages, independent of the other clients. The simulator uses timeouts and delays to set the wait-time between events. Consequently, rather than using the timestamps for the exact time a message is sent, the time period before a client schedules a new message is determined relative to the last message sent. Delays are generated as the time difference between every two consecutive messages.

5.3 Message Scheduling

When the client schedules a message, the message is added to the client's buffer. Another client process sends out messages with delays sampled from an exponential distribution with a predefined rate. The process either sends out a real message by extracting it from the buffer (in case a real message has been scheduled and the buffer is not empty), or by sending out a dummy message. This type of decoy traffic is referred to as *loop traffic* [35]. Either way, a message is sent at the predefined rate, be it real or a dummy.

At the start of the simulation, each client is given the messages they will schedule throughout the whole course of the simulation. The messages are stored in an array containing tuples of the already generated `Message` instance and the time to wait after the last message was sent before scheduling the new one. Each client schedules messages by looping through the message array, scheduling a timeout to wait a certain amount of time specified by the delay of the message before scheduling the message and proceeding to the next one.

Special care needs to be taken for the first message a client schedules to ensure it is not sent out immediately after the start of the simulation. The wait-time of the first message for each client is determined as the difference of the timestamp of the client's first message and the very first message in the entire message log.

5.4 Termination Condition

The simulation runs until a termination condition has been satisfied. The termination condition is satisfied after all real packets had been sent. Once all these real packets are received, the termination event is triggered, which completes the simulation. The test mode only supports the simulation of real packets by a single sender which is allocated control over the termination of the entire simulation. This would not fit the

transcript mode as it does not extend to multiple senders and a varying number of packets depending on the message log file. Instead, an additional flag was added to the `Message` class, which is set to true if that message is the last one in the log file. When the final message is sent, the termination condition is satisfied, which signifies that no more real packets will be transported over the network. Only once all real messages are received, the termination event is triggered and the simulation is completed.

5.5 Anonymity and Unlinkability Measurement

The test mode of the simulator generates 'real' messages split into packets. The system simulates a fixed number of real packets, specified by the simulation configuration file and calculates their entropy.

For transcript mode, the number of real packets depends on the message payload size and the number of messages as stored in the log file. When generating packets, each packet is given a unique id which allowed tracking the packet and recording its entropy upon arrival at the last mix node before it is forwarded to the recipient. The total entropy is calculated by averaging out the entropy of all real packets. This allowed determining the entropy on the entire set of real network traffic.

The E2E unlinkability is calculated by randomly selecting two clients (senders) at the start of the simulation and recording the probability of each received packet originating from the first sender, the second or neither.

Chapter 6

Evaluation

This chapter reports the experimental results of the evaluation of the Email Data Generator both with and without applying the private degree distribution approximation algorithm. The chapter analyses the privacy-utility trade-off of using the differentially private degree distribution when generating the synthetic social graph. We further report results on how perturbing the shape of the network affects communication security and conclude that communication anonymity does not depend on the topology of the social graph but on the temporal patterns of the correspondence of accounts. The chapter begins by presenting the settings of the evaluation.

6.1 Datasets and Experimental Settings

This section provides the settings details under which we conducted our experiments. The section can be used as a reference for the potential replication of the experiments, the results of which are presented in the Sections 6.2 to 6.4.

6.1.1 Datasets

As previously mentioned, the implementation of the Email Data Generator is based on the Enron Email dataset and as such we recognise it might not extend to other email communication datasets with different social-graph characteristics and temporal sending patterns. Therefore, we have chosen to evaluate the data generation process on the Sendmail Log dataset.

The preprocessed Sendmail Log dataset contains email transactions over the course of two months for a total of 3848 accounts. Such large timespan and graph size make it infeasible to run the Loopix simulator on the whole dataset. The simulator generates a separate process for each account, which makes the simulation computationally heavy should the network consist of a large number of accounts. We chose to limit the time frame of the simulation by extracting the email communications within a single week. We believe the timespan of a week is small enough to be simulated but large enough to capture the hourly email sending patterns when synthetic data is being generated.

This allowed scaling down the dataset to email transactions of 124 accounts over the course of a single week. The same reduced dataset was used to generate synthetic data with and without applying the differentially private degree approximation. The reduced real dataset and the synthetic ones generated from it were only used for the Loopix simulation. The rest of the evaluation is based on the entire processed Sendmail Log dataset.

Where no θ and ε values are explicitly stated and the data is simply referred to as 'synthetic', the original degree distribution has been used for the graph generation.

6.1.2 Choosing ε and θ

Privacy has been rigorously formalised through the introduction of the privacy budget ε. The value of the privacy parameter ε quantifies how much importance we attribute to minimising the privacy loss when producing a private social graph degree distribution. However, there is no rigorous method for choosing the exact value of ε [18]. Stringent privacy needs ask for the value of ε to be less than 1. However, by relaxing privacy requirements, ε could take values of up to 10.
Small values of ε provide higher guarantees, however, require more random noise to be added to the degree distribution of the social graph, thus distorting the original distribution of the graph and potentially resulting in a completely new graph topology. High ε values have smaller impact on the original degree distribution by reducing the amount of distortion to the degree distribution but would fail to hide the social graph. To cover both of these extremes and observe the impact of the privacy budget on the utility of the private social graph release, we have chosen $\varepsilon \in [0.1, 0.25, 0.5, 1, 1.5, 2, 3, 5, 7.5, 10]$.

The degree distribution projection algorithm bounds the largest degree in the graph to predefined value θ to reduce the amount of noise which has to be added to the resultant θ-bound degree distribution. The optimal choice of θ depends on the degree distribution of the original graph and the choice of ε. Larger θ will allow higher degrees to be present in the private degree distribution but will add more noise, thus heavily distorting the distribution. In the case of both datasets that we have been working with, the social network is scale-free, which means that a big proportion of the nodes are of small degree. In this situation, a small value θ will still be able to preserve a lot of the nodes' original degree because of the low number of high-degree nodes, while maintaining a low noise level.
The social graph of the processed Sendmail Log dataset has 3848 nodes, with mean degree 4.92 and standard deviation 32.34. Table 6.1 presents the proportion of nodes with degree smaller or equal to some upper bound. There are very few nodes with large

θ	4	8	16	25	50	75	100
$\frac{\lvert V^{d\leq\theta}\rvert}{\lvert V\rvert}$	0.8613	0.9208	0.9578	0.9741	0.9890	0.9930	0.9947

Table 6.1: Sendmail Log Graph: Partial Cumulative Distribution

degree. The proportion of nodes with degree smaller than the mean is 0.86. Therefore, setting the upper bound $\theta = 4$ would preserve a large proportion of the original degrees

of the graph while adding a small amount of noise to the truncated degree distribution. With regards to the degree distribution of the graph, we explore a wide range of upper bounds without setting the upper bound too high: $\theta \in [4, 8, 16, 25, 50, 100]$.

As mentioned in the previous section, the reduced dataset contains a significantly smaller account set consisting of 124 accounts. The average degree is 7.96 and the maximum is 45. Therefore, we define a separate set of upper bounds for the reduced dataset which was used for the Loopix simulation: $\theta \in [4, 8, 16, 25]$.

6.1.3 Loopix Simulator Setup

We ran the Loopix Simulator with fixed mixing delay $\mu = 1ms$. The delays of all messages at each mix node are sampled from an exponential distribution with parameter μ. Each client has a sending rate of $10ms$ (that is the rate at which the buffer is checked for scheduled messages or a dummy message is sent out). The network has 9 mix nodes in a stratified topology with 3 layers and 3 mix nodes in each layer, as described in [35]. Every email message is contained in a single `Sphinx` packet. Cover traffic has been disabled (this does not include loop traffic).
A total of 42 simulations were run on a 2.80 *GHz*, 16GB *RAM* Windows machine. Each simulation took an average of 9.5 hours. Each simulation setting was run a single time due to limited time constraints.

6.2 Statistical Evaluation of Synthetic Data

This section presents the statistical evaluation of the synthetic data in comparison to the real data. We use the clustering coefficient as a measure of the degree to which the nodes in the graph are clustered together. A clique is a subset of vertices V' of graph G such that the subgraph induced by V' is complete (there is an edge between any two distinct vertices in V'). The clique number of a graph is the largest number of vertices in the original graph which form a clique. The clique number and clustering coefficient are used to compare the synthetic graph to the original.

The graph generation algorithm constructs edges by randomly selecting two vertices which have not reached their prescribed degree. It does not preserve the clustering properties of the original graph as seen by the low clustering coefficient for the synthetically generated graph.

	d_{max}	d_{mean}	d_{std}	Clustering Coefficient	Clique Number	Power Law Exponent
real	1351	4.92	32.33	0.2879	12	2.23
synthetic	933	5.21	35.08	0.1003	5	2.45

Table 6.2: Real and Synthetic Social Graph Comparison

By fitting the original degree distribution, we are able generate a scale-free synthetic

graph. Both graphs follow a Power Law distribution at their tails with close Power Law exponent values.

Table 6.3 outlines the message characteristics of the original and synthetic datasets. A message is a single email generated by one account to one or potentially more recipients. For the generation of timestamps, we extract the hourly sending rates over the course of a week of the real data. To explore how well those rates are preserved for the synthetic email dataset, we construct the normalised cumulative distribution of hourly sending rates for both synthetic and real email messages. The maximum cumulative KS-distance is calculated to determine how closely the synthetic rates follow those of the original data.

The synthetic data preserves the email sending rates of the original data, which is indicated by the small KS-distance. Additionally, the preserved sending rates result in a total number of generated messages within ± 0.05 range of the size of the original message set. However, the maximum number of email messages generated by a single account is considerably lower compared to the most active account in the real data. This can be attributed to the rarity of highly active accounts.

	$\|M\|$	$\|M\|_{weekly,avg}$	$\|M_i\|_{avg}$	$\|M_i\|_{max}$	Weekly-Hourly sending rate KS-distance
real	87412	9712.4	29.08	9684	0.0
synthetic	85245	9471.7	23.58	5936	0.05

Table 6.3: Real and Synthetic Social Message Comparison

6.3 Synthetic Private Social Graph

6.3.1 Random Cauchy Noise

Table 6.4 shows the scale parameter of Cauchy noise for different upper bound θ and privacy parameter ε values. As described in Section 4.2.2, randomly generated noise is added to the degree frequency for all degrees $d \in [1, \ldots, \theta]$. The noise decreases as the privacy guarantees are relaxed by increasing the privacy budget. Conversely, as θ increases, more noise must be added to 'blend' the large-degree nodes in the degree distribution, resulting in a large scale parameter.

		ε									
		0.1	0.25	0.5	1.0	1.5	2.0	3.0	5.0	7.5	10.0
θ	4	99.99	37.53	16.87	6.82	3.68	2.233	0.97	0.25	0.08	0.06
	8	205.68	75.76	33.01	12.53	6.34	3.61	1.39	0.27	0.05	0.03
	16	420.36	153.12	65.50	23.97	11.70	6.42	2.29	0.39	0.06	0.03
	25	660.66	239.61	101.77	36.71	17.66	9.55	3.31	0.54	0.07	0.04
	50	1336.27	482.72	203.66	72.50	34.41	18.37	6.21	0.95	0.11	0.03
	100	2720.67	980.81	412.38	145.80	68.73	36.452	12.15	1.82	0.21	0.07

Table 6.4: Random Cauchy Noise Scale with respect to θ and ε

6.3.2 Evaluation

As a result of the private degree distribution approximation, the social graph is no longer scale-free. While this is to be expected due to the removal of nodes with large degrees, it results in a significant change in the structure of the graph.

Taking into consideration the nature of the original graph, smaller θ would preserve a large proportion of the node's original degrees, while keeping the amount of random noise low. However, a small maximum degree limits the number of edges which can be generated within the graph. This is evident from the low Preserved Edge Ratio (PER) measure for projected graphs with θ equal to 4 and 8.
Strong privacy guarantees, where the privacy budget is below 1, and a large upper bound result in significant perturbation of the original graph's degree distribution. Adding large amounts of noise creates an arbitrary degree distribution, with a considerably higher average degree in comparison with the original graph (Table 6.5 and 6.6). Consequently, extreme PER values are observed for $\theta \geq 50$ and a tight privacy budget $\varepsilon \leq 1$.

We have compared the node-DP graph for various ε values. The degree distribution of the original graph and projected graphs are converted to a θ-Cumulative Histogram. KS-distance is used to determine the closeness of the node-DP histogram to the original one. As expected, relaxing the privacy bound leads to very little noise being added to the truncated original degree distribution. The result is low KS-distance smaller than 0.1 and a θ-bound degree distribution which is almost identical to the real θ-bound distribution.

θ	ε	d_{max}	d_{mean}	d_{std}	Clustering Coefficient	Clique Number
4	0.1	4	2.00	0.09	0.0007	3
	0.25	3	1.42	0.68	0.0009	3
	0.5	4	1.54	0.85	0.0003	3
	1.0	4	1.50	0.82	0.0002	3
	1.5	4	1.50	0.82	0.0002	3
	2.0	4	1.48	0.79	0.0007	3
	3.0	4	1.44	0.75	0.0004	3
	5.0	4	1.53	0.87	0.0002	3
	7.5	4	1.49	0.82	0.0006	3
	10.0	4	1.48	0.79	0.0004	3
8	0.1	8	5.08	2.35	0.0028	3
	0.25	8	1.83	1.75	0.0012	3
	0.5	8	2.04	1.75	0.0013	3
	1.0	8	1.77	1.53	0.0012	3
	1.5	8	2.10	1.82	0.0011	3
	2.0	8	1.72	1.27	0.0004	3
	3.0	8	1.80	1.42	0.0007	3
	5.0	8	1.81	1.40	0.0004	3
	7.5	8	1.79	1.43	0.0002	3
	10.0	8	1.78	1.39	0.0003	3
16	0.1	16	5.81	4.18	0.0038	3
	0.25	16	5.54	3.84	0.0034	3
	0.5	16	3.66	4.05	0.0028	3
	1.0	16	2.62	2.97	0.0019	3
	1.5	16	2.79	3.22	0.0023	3
	2.0	15	3.46	4.46	0.0030	3
	3.0	16	3.99	4.57	0.0033	3
	5.0	16	2.24	2.52	0.0019	3
	7.5	16	2.16	2.48	0.0009	3
	10.0	16	2.19	2.42	0.0019	3

Table 6.5: Private Graph Characteristics for $\theta = [4, 8, 16]$

The random generation of edges while constructing the synthetic graph, means the edges are arbitrarily generated without the preservation of clusters. This explains the low clustering coefficient and clique number of the synthetic graphs. However, this is inflicted by the graph generation algorithm and is not the result of the differentially private graph projection.

Ultimately, if we wish to hide the structure of the original graph by using a private degree distribution approximation, we have to tolerate a considerable change in the original graph's degree distribution.

θ	ε	d_{max}	d_{mean}	d_{std}	Clustering Coefficient	Clique Number
25	0.1	24	7.74	5.77	0.0049	3
	0.25	25	11.60	9.53	0.0080	4
	0.5	25	12.26	10.31	0.0088	4
	1.0	22	2.49	3.47	0.0022	3
	1.5	21	3.61	4.74	0.0034	3
	2.0	25	3.87	5.15	0.0039	3
	3.0	25	2.57	3.59	0.0025	3
	5.0	24	2.42	3.14	0.0023	3
	7.5	25	2.55	3.45	0.0017	3
	10.0	25	2.39	3.25	0.0019	3
50	0.1	50	43.28	13.68	0.0232	4
	0.25	50	26.62	16.55	0.0165	4
	0.5	49	21.55	12.42	0.0128	4
	1.0	50	27.59	16.64	0.0170	4
	1.5	48	11.25	14.08	0.0119	4
	2.0	49	6.49	11.30	0.0085	3
	3.0	49	5.91	10.44	0.0096	3
	5.0	50	4.72	9.89	0.0091	3
	7.5	50	2.86	5.40	0.0045	3
	10.0	47	2.97	5.08	0.0040	3
100	0.1	100	54.87	24.72	0.0311	5
	0.25	98	53.01	27.98	0.0313	5
	0.5	97	59.36	30.34	0.0346	5
	1.0	100	54.54	32.91	0.0334	5
	1.5	100	34.63	31.01	0.0263	5
	2.0	100	37.25	24.19	0.0236	4
	3.0	100	15.56	25.78	0.0234	4
	5.0	100	6.21	14.96	0.0161	3
	7.5	98	3.59	8.25	0.0074	3
	10.0	98	3.46	8.27	0.0055	3

Table 6.6: Private Graph Characteristics for $\theta = [25, 50, 100]$

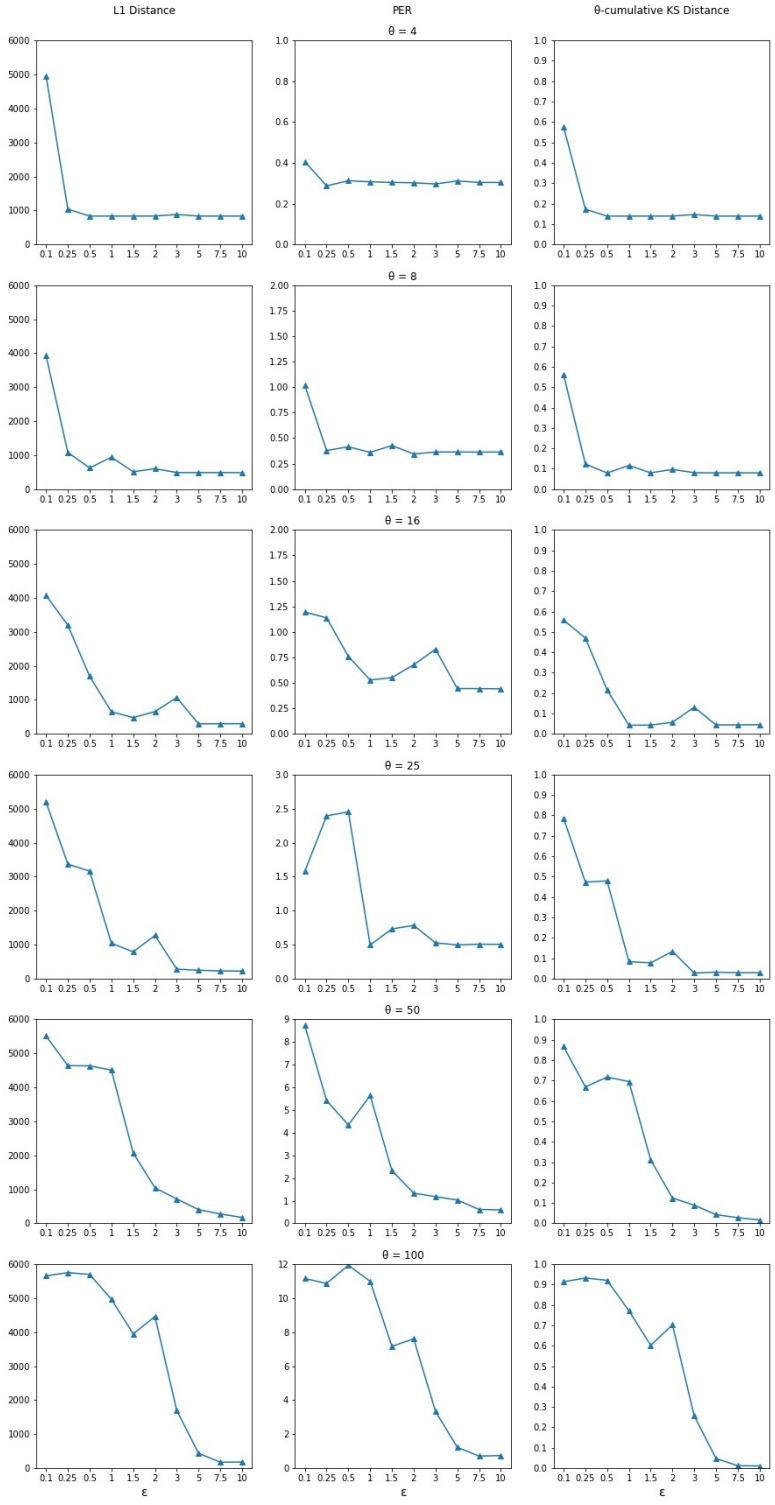


Figure 6.1: L1 Distance, PER and KS-Distance for private graphs $\theta = [4, 8, 16, 25, 50, 100]$ and $\varepsilon = [0.1, 0.25, 0.5, 1, 1.5, 2, 3, 5, 7.5, 10]$

6.4 Simulation

Here we present the results of the simulation experiment conducted for the real and synthetic datasets. The role of these experiments is two-fold: it allows us to compare the anonymity level of the synthetic datasets in comparison to the real dataset, as well as evaluate the impact of generating a social graph from a private degree distribution on the communication anonymity of the simulation.

There is very little fluctuation in the recorded entropy for the synthetic datasets in comparison with the original dataset (Tables 6.7 and 6.8). The entropy of each packet is computed incrementally at each mix node on its path from the sender to the recipient. Its value depends on the state of the mix node's pool and on the entropy at the previous pool. Therefore, the entropy is highly dependent on the rate at which messages arrive at the mix node's pool and the preconfigured mixing delay. The higher the rate, the more messages are mixed in the pool at the same time, thus increasing anonymity. Similarly, configuring a higher mixing delay means more messages are accumulated at the mix node's pool, leading to a larger anonymity set and increased entropy.

Both the original and synthetic datasets follow the same sending rates, which is the main determinant of the system's anonymity level. Therefore, the structure of the underlying communication graph, does not influence the security of the actual transactions between nodes. The little variation of the entropy of each simulation confirms this. In this sense, the entropy measure provides an insight of how well the email generation and sending rates of the original dataset are reflected in the synthetic data.

The E2E unlinkability - the expected difference in the likelihood that a message originated from one sender in comparison to another, varies significantly for different simulation settings. However, no consistent pattern can be seen for different θ and ε values and more simulation experiments need to be conducted to make any conclusive statement, time permitting.

	entropy	E2E unlinkability
real	6.152	11.86
synthetic	6.149	10.17

Table 6.7: Loopix Simulation Results for Real and Synthetic data

θ	ε	entropy	E2E unlinkability
4	0.1	6.140	8.36
	0.25	6.149	11.05
	0.5	6.141	10.41
	1	6.144	9.24
	1.5	6.141	8.99
	2	6.144	9.75
	3	6.139	10.32
	5	6.140	11.11
	7.5	6.138	8.29
	10	6.158	7.31
8	0.1	6.151	8.60
	0.25	6.145	8.83
	0.5	6.151	10.78
	1	6.150	12.03
	1.5	6.146	9.04
	2	6.138	12.08
	3	6.141	10.75
	5	6.136	13.19
	7.5	6.150	11.22
	10	6.144	8.85
16	0.1	6.144	8.82
	0.25	6.138	9.68
	0.5	6.153	9.89
	1	6.149	10.63
	1.5	6.146	9.94
	2	6.145	8.94
	3	6.152	10.53
	5	6.141	9.34
	7.5	6.144	9.30
	10	6.155	11.38
25	0.1	6.135	11.17
	0.25	6.145	10.01
	0.5	6.146	10.58
	1	6.153	10.32
	1.5	6.139	7.10
	2	6.140	10.38
	3	6.155	10.49
	5	6.135	8.57
	7.5	6.144	9.42
	10	6.145	13.00

Table 6.8: Loopix Simulation Results

6.5 Results

Our statistical evaluation concludes that the generation workflow successfully preserves the email generation rates and the scale-free nature of the network graph (when no private degree approximation is applied). However, we identify an inadequacy in the graph generation algorithm which only maintains the degree distribution of the real graph and fails to preserve its structure in terms of clustering. This shortcoming of the implementation will be discussed further in Section 7.2.

The evaluation of the node-DP graph concludes that strong privacy guarantees over the degree distribution of a graph can only be achieved if we allow a substantial shift from the original distribution.

The simulation results indicate that the synthetic data performs similarly to the real data in terms of entropy as a measure of the system's anonymity level. This allows us to conclude that the network's topology does not play a role in determining how secure the email transactions over the MixNet network are. We instead explain the similar entropy measure by the sending rates for the synthetic datasets which had been preserved from the original data.

Chapter 7

Conclusion

This chapter provides a summary of the conducted work regarding the implementation and evaluation of the Email Dataset Generator. It outlines limitations and opportunities for future work with respect to those.

7.1 Summary

This report presented the Email Dataset Generator, a framework which infers the statistical characteristics of an email log and generates synthetic email traffic within a network. The project further implements a node differentially private degree distribution approximation, which allows hiding the original degree distribution by removing large-degree nodes and perturbing the resultant distribution. We implement a transcript mode of the Loopix MixNet simulator which allowed us to evaluate the privacy leakage of the original and synthetic datasets through email traffic simulation. We further provide a quantified evaluation of the effects of hiding the graph's degree distribution on the generation of the synthetic graph and email transactions.

Based on the results of our evaluation we conclude that we cannot provide strict privacy guarantees over the degree distribution of a communication network without significant perturbation to the distribution and therefore to the overall structure of the original scale-free network. The results of our simulation demonstrate that the synthetic data performs similarly to the real data in terms of the anonymity level of the email transactions. We attribute this to the email generation rates which are preserved from the real dataset to the synthetic ones.

The transcripted simulation of Loopix was only implemented to assist the experiments and evaluation process of this project. However, we believe it constitutes a significant contribution to the open-source simulator by enabling a wide range of experiments to be conducted.

7.2 Limitations

A major weakness of the generation process lies within the graph construction algorithm. The graph generation algorithm presented in Section 4.2.3 constructs a synthetic graph from a degree distribution alone. While this allows for a differentially private graph to be generated by simply perturbing the degree distribution, it fails to capture the structure of the graph in terms of cliques and clusters. In the specific case of the two datasets we worked with, clusters often correspond to teams and groups within a greater organisation or community. As such, they are an important part of a social graph and should be reflected in an accurate synthetic representation. However, any metric used for the generation of a synthetic graph (such as number and size of cliques or shortest path between nodes) would have to be released under a differentially private algorithm if differentially private guarantees were to be enforced.

Here, we choose to discuss the decision to use the Enron dataset as a reference when designing the statistical generation process. While the dataset provides a valuable reference to determine which properties of the data to be statistically inferred, it inevitably led to making general assumptions which might not apply to other datasets. This reasoning is exactly what motivated the decision to evaluate the Generator and conduct simulations on the Sendmail Log dataset, allowing us to determine how well the generation extended to other datasets.

The generation process makes a number of assumptions inherent to the Enron dataset. This makes the generation well suited for email traffic within an organisation or professional institution, where weekly email sending patterns can be observed, correlated to business working hours. Both datasets have a relatively low hourly email sending rate and no obvious patterns of email generation within a single hour, which motivated the decision to model the temporal email sending patterns using hourly sending rates within a single week. However, more fine-grained modelling will be needed for high traffic rates or a small dataset with traffic contained within a short period of time.

Additionally, the quality of the generated data is highly dependent on the evaluation criteria it is made subject to. Therefore, pitfalls and limitations of the evaluation process can either overestimate or underestimate the significance of the obtained results. The evaluation fails to examine if any correlation exists between the degree of nodes and their activity level (whether accounts which send emails to a lot of other accounts also actively communicate by sending a large number of emails) and whether such correlation is preserved in the synthetic dataset.

7.3 Future work

The limitations presented in the previous section create room for future work. Main focus of improvement would be the graph generation algorithm. Parametric models such as Forest Fire [23] and Nearest Neighbours [41] iteratively construct a synthetic graph by considering both the node degrees and the shortest path length between pairs of nodes. They aim to minimise the distance between the real and the synthetic graphs.

The 2.5K-graph model [13] approximates the joint degree distribution and the clustering coefficient of the real graph when generating a synthetic one. Those models would be more successful at reflecting the original topology onto the synthetic graph and constitute key modifications that could have been applied, time permitting.

Bibliography

[1] Eric Allman. *Sendmail Installation and Operation Guide For Sendmail*, 04 2000.

[2] Karolyn O. Babalola, Otis B. Jennings, Esteban Urdiales, and James A. DeBardelaben. Statistical methods for generating synthetic email data sets. In *2018 IEEE International Conference on Big Data (Big Data)*, pages 3986–3990, 2018.

[3] Joseph Blitzstein and Persi Diaconis. A sequential importance sampling algorithm for generating random graphs with prescribed degrees. *Internet Mathematics*, 6(4):489 – 522, 2010.

[4] Federal Energy Regulatory Commission. Enron email dataset. `https://www.cs.cmu.edu/~enron/`.

[5] Germán Creamer, Ryan Rowe, Shlomo Hershkop, and Salvatore J. Stolfo. *Segmentation and Automated Social Hierarchy Detection through Email Network Analysis*, page 40–58. Springer-Verlag, Berlin, Heidelberg, 2009.

[6] Wei-Yen Day, Ninghui Li, and Min Lyu. Publishing graph degree distribution with node differential privacy. In *Proceedings of the 2016 International Conference on Management of Data*, SIGMOD '16, page 123–138, New York, NY, USA, 2016. Association for Computing Machinery.

[7] Claudia Diaz, Stefaan Seys, Joris Claessens, and Bart Preneel. Towards measuring anonymity. volume 2482, pages 54–68, 04 2002.

[8] Cynthia Dwork. Differential privacy. In Michele Bugliesi, Bart Preneel, Vladimiro Sassone, and Ingo Wegener, editors, *Automata, Languages and Programming*, pages 1–12, Berlin, Heidelberg, 2006. Springer Berlin Heidelberg.

[9] Cynthia Dwork, Frank McSherry, Kobbi Nissim, and Adam Smith. Calibrating noise to sensitivity in private data analysis. In Shai Halevi and Tal Rabin, editors, *Theory of Cryptography*, pages 265–284, Berlin, Heidelberg, 2006. Springer Berlin Heidelberg.

[10] Khaled El Emam, David L. Buckeridge, Robyn Tamblyn, Angelica Neisa, Elizabeth Jonker, and Aman Verma. The re-identification risk of canadians from longitudinal demographics. *BMC Medical Informatics and Decision Making*, 11:46 – 46, 2011.

[11] Yaniv Erlich and Arvind Narayanan. Routes for breaching and protecting genetic privacy. *Nature Reviews Genetics*, 15(6):409–421, June 2014. Funding

Information: Y.E. is an Andria and Paul Heafy Family Fellow and holds a Career Award at the Scientific Interface from the Burroughs Wellcome Fund. This study was supported in part by a US National Human Genome Research Institute grant R21HG006167, and by a gift from C. Stone and J. Stone. The authors thank D. Zielinski and M. Gymrek for comments.

[12] Fabrizio Gabbiani and Steven James Cox. *Mathematics for Neuroscientists*. Elsevier Science Technology, Saint Louis, 1 edition.

[13] Minas Gjoka, Maciej Kurant, and Athina Markopoulou. 2.5k-graphs: from sampling to generation, 2012.

[14] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In Z. Ghahramani, M. Welling, C. Cortes, N. Lawrence, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems*, volume 27. Curran Associates, Inc., 2014.

[15] UK Government. Data protection act of 2018, 2018.

[16] J. S. Hardin, G. Sarkis, and P. C. URC. Network analysis with the enron email corpus. *Journal of Statistics Education*, 23(2), 2015.

[17] Michael Hay, Chao Li, Gerome Miklau, and David Jensen. Accurate estimation of the degree distribution of private networks. In *2009 Ninth IEEE International Conference on Data Mining*, pages 169–178, 2009.

[18] Justin Hsu, Marco Gaboardi, Andreas Haeberlen, Sanjeev Khanna, Arjun Narayan, Benjamin C. Pierce, and Aaron Roth. Differential privacy: An economic method for choosing epsilon. In *2014 IEEE 27th Computer Security Foundations Symposium*, pages 398–410, 2014.

[19] Shiva Prasad Kasiviswanathan, Kobbi Nissim, Sofya Raskhodnikova, and Adam Smith. Analyzing graphs with node differential privacy. In Amit Sahai, editor, *Theory of Cryptography*, pages 457–476, Berlin, Heidelberg, 2013. Springer Berlin Heidelberg.

[20] Dogan Kesdogan, Jan Egner, and Roland Büschkes. Stop- and- go-mixes providing probabilistic anonymity in an open system. In David Aucsmith, editor, *Information Hiding*, pages 83–98, Berlin, Heidelberg, 1998. Springer Berlin Heidelberg.

[21] Diederik P. Kingma and Max Welling. Auto-Encoding Variational Bayes. In *2nd International Conference on Learning Representations, ICLR 2014, Banff, AB, Canada, April 14-16, 2014, Conference Track Proceedings*, 2014.

[22] Bryan Klimt and Yiming Yang. The enron corpus: A new dataset for email classification research. In Jean-François Boulicaut, Floriana Esposito, Fosca Giannotti, and Dino Pedreschi, editors, *Machine Learning: ECML 2004*, pages 217–226, Berlin, Heidelberg, 2004. Springer Berlin Heidelberg.

[23] Jure Leskovec, Jon Kleinberg, and Christos Faloutsos. Graphs over time: Densification laws, shrinking diameters and possible explanations. In *Proceedings of*

the Eleventh ACM SIGKDD International Conference on Knowledge Discovery in Data Mining, KDD '05, page 177–187, New York, NY, USA, 2005. Association for Computing Machinery.

[24] Yang Li, Michael Purcell, Thierry Rakotoarivelo, David B. Smith, Thilina Ranbaduge, and Kee Siong Ng. Private graph data release: A survey. *CoRR*, abs/2107.04245, 2021.

[25] Bradley Malin and Latanya Sweeney. How (not) to protect genomic data privacy in a distributed network: Using trail re-identification to evaluate and design anonymity protection systems. *J. of Biomedical Informatics*, 37(3):179–192, jun 2004.

[26] Steve Martin, Blaine Nelson, Anil Sewani, Karl Chen, and Anthony D. Joseph. Analyzing behavioral features for email classification. In *CEAS*, 2005.

[27] Sumit Mukherjee, Yixi Xu, Anusua Trivedi, and Juan Lavista Ferres. privgan: Protecting gans from membership inference attacks at low cost to utility. *CoRR*, abs/2001.00071, 2020.

[28] Arvind Narayanan and Vitaly Shmatikov. Robust de-anonymization of large sparse datasets. In *2008 IEEE Symposium on Security and Privacy (sp 2008)*, pages 111–125, 2008.

[29] M. E. J. Newman. The structure and function of complex networks. *SIAM Review*, 45(2):167–256, 2003.

[30] Council of European Union. Regulation on the protection of natural persons with regard to the processing of personal data and on the free movement of such data, May 2016.

[31] Vijay Pandurangan. On taxis and rainbows: Lessons from nyc's improperly anonymized taxi logs, Jun 2014.

[32] Raghu Pasupathy. Generating nonhomogeneous poisson processes. 2011.

[33] Kelly Peterson, Matt Hohensee, and Fei Xia. Email formality in the workplace: A case study on the Enron corpus. In *Proceedings of the Workshop on Language in Social Media (LSM 2011)*, pages 86–95, Portland, Oregon, June 2011. Association for Computational Linguistics.

[34] Haoyue Ping, Julia Stoyanovich, and Bill Howe. Datasynthesizer: Privacy-preserving synthetic datasets. In *Proceedings of the 29th International Conference on Scientific and Statistical Database Management*, SSDBM '17, New York, NY, USA, 2017. Association for Computing Machinery.

[35] Ania M. Piotrowska, Jamie Hayes, Tariq Elahi, Sebastian Meiser, and George Danezis. The loopix anonymity system. *CoRR*, abs/1703.00536, 2017.

[36] H. A. Piwowar and T. J. Vision. Data reuse and the open data citation advantage. *PeerJ*, 1, 2013.

[37] C. E. Shannon. A mathematical theory of communication. *The Bell System Technical Journal*, 27(3):379–423, 1948.

[38] Qiuye Sun, Xuan Liu, Genlang Chen, Shiting Wen, and Guanghui Song. An improved sanitization algorithm in privacy-preserving utility mining. *Mathematical Problems in Engineering*, 2020, 2020.

[39] Latanya Sweeney. k-anonymity: A model for protecting privacy. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, 10, 05 2012.

[40] NYC Taxi and Limousine Commission. Tlc trip record data.

[41] Alexei Vázquez. Growing network with local rules: Preferential attachment, clustering hierarchy, and degree correlations. *Phys. Rev. E*, 67:056104, May 2003.

[42] Duncan J. Watts and Steven H. Strogatz. Collective dynamics of 'small-world' networks. *Nature*, 393(6684):440–442, 1998.

[43] Jinsung Yoon, James Jordon, and Mihaela van der Schaar. PATE-GAN: Generating synthetic data with differential privacy guarantees. In *International Conference on Learning Representations*, 2019.

[44] Jun Zhang, Graham Cormode, Cecilia M. Procopiuc, Divesh Srivastava, and Xiaokui Xiao. Privbayes: Private data release via bayesian networks. *ACM Trans. Database Syst.*, 42(4), oct 2017.

[45] Yingjie Zhou, Mark Goldberg, Malik Magdon-ismail, and William A. Wallace. Strategies for cleaning organizational emails with an application to enron email dataset.