CS 860: Algorithms for Private Data Analysis Fall 2020

Lecture 1 — Some Attempts at Data Privacy

Prof. Gautam Kamath *Scribe: Gautam Kamath*

This course will be largely focused on private data analysis, under the definition of differential privacy. But before we get to that, let's talk a bit about a few failures in data privacy, and why we need rigorous notions of privacy to ensure we don't leak sensitive information.

NYC Taxicab Data

Back in 2014, the NYC Taxi & Limo Commission was quite active on Twitter, sharing visualizations of a number of taxi usage statistics. This quickly caught the eye of several Internet users, who inquired about the source of this data. The Taxi & Limo Commission replied that this data is available, but one must file a Freedom of Information Law (FOIL) request. Freedom of Information laws allow for citizens to request data from certain government organizations. In the US, this is governed at the federal level by the Freedom of Information Act (FOIA). In Canada, similar laws exist, including the Freedom of Information and Protection of Privacy Act (FIPPA) and the Municipal Freedom of Information and Protection of Privacy Act (MFIPPA) Act. Chris Whong filed a FOIL request and released the dataset publicly online [Who14]. He also documented his experiences filing the request, which is an amusing read, but not our focus today.

The dataset he obtained consists of all taxi fares and trips in NYC during 2013 – a total of 19 GB worth of data. This should immediately set off some alarm bells: indeed, if this dataset identifies which drivers are giving these rides, then we would have information about the location and income of every taxi driver in New York City! This is information we generally consider to be sensitive, and a taxi driver might prefer to keep private. As you might expect, the Commission attempted to obscure this information, using a form of anonymization.

A typical row in the trips dataset looks like the following:

```
6B111958A39B24140C973B262EA9FEA5,D3B035A03C8A34DA17488129DA581EE7,VTS,5,,2013-12-03
15:46:00,2013-12-03 16:47:00,1,3660,22.71,-73.813927,40.698135,-74.093307,40.829346
```

These fields are:

```
medallion, hack_license, vendor_id, rate_code, store_and_fwd_flag, pickup_datetime,
dropoff_datetime, passenger_count, trip_time_in_secs, trip_distance, pickup_longitude,
pickup_latitude, dropoff_longitude, dropoff_latitude
```

While most of these fields are rather self-explanatory, such as the time and location fields, the first two are of primary interest to us. In particular, they indicate a taxi driver's medallion and license number. However, the standard format for these fields is rather different than what is provided – it appears that the dataset has been somehow anonymized, to mask these values.

Upon inspection, Jason Hall posted on Reddit [Hal14] that someone with the medallion number `CFCD208495D565EF66E7DFF9F98764DA` had a number of unusually profitable days, making much more than a taxi driver could hope to make. Vijay Pandurangan dug a bit deeper on this, and

made the following discovery[Pan14]:

```
gautam@gautam-ThinkPad-P51:~$ echo -n 0 | md5sum
cfcd208495d565ef66e7dff9f98764da  -
```

This demonstrates that taking the MD5 hash of the string "0" gives the identifier mentioned above. Pandurangan hypothesized that this identifier corresponded to instances when the medallion number wasn't available, but he took this as a hint: all the medallion and license numbers were simply the plaintext hashed via MD5. As both these identifiers are only a few characters long, he was able to compute the MD5 hashes of all possibilities and obtain the pre-hashing values for the entire dataset. Using other publicly available data, he was further able to match these with driver names, thus matching real-life identities with incomes and locations: a massive privacy violation!

One might immediately think of some ways to avoid this issue. For instance, rather than hashing the true medallion and license numbers, why not make up entirely new and arbitrary identifiers? However, this is still susceptible to privacy violations: suppose you took a ride with a driver, and noted down the time and location of your trip. Then by referencing the data afterwards, you can discover the driver's identifiers and thus their income and location history.

This type of attack using *side-information* goes beyond revealing information about only the drivers. Suppose a co-worker says they're going home after work – as you wave them goodbye, you record the pickup time and location. If you later reference the dataset with this information, you can discover their home address (as well as whether they're a generous tipper or not).

The bottom line: privacy is hard. And we're about to see a whole bunch of other examples.

The Netflix Prize

Another case study of data anonymization gone wrong is the Netflix Prize competition. Netflix is a very data-driven and statistically-minded company: many of their hit TV shows are conceived based on user data, and their fabled recommendation algorithm is tuned to optimize user engagement. Between 2006 and 2009, they hosted a contest, challenging researchers to improve their recommendation engine. The grand prize was a highly-publicized US$1,000,000, claimed by a team named BellKor's Pragmatic Chaos, based on matrix factorization techniques.

In order to help teams design their strategies, Netflix provided a training dataset of user data. Each datapoint consisted of an (anonymized) user ID, movie ID, rating, and date. Netflix assured users that the data was appropriately de-anonymized to protect individual privacy. Indeed, the Video Privacy Protection Act of 1988 requires them to do this. One's media consumption history is generally considered to be sensitive or private information, as one might consume media associated with certain minority groups (including of a political or sexual nature).

Unfortunately, Narayanan and Shmatikov demonstrated that this naïve form of anonymization was insufficient to preserve user privacy[NS08]. Their approach is illustrated in Figure 1. They took the dataset provided by Netflix, and cross-referenced it with public information from the online movie database IMDb, which contains hundreds of millions of movie reviews. In particular, they tried to match users between the two datasets by finding users who gave similar ratings to a movie at similar times. While the Netflix data was de-identified, the IMDb data was not, and a review

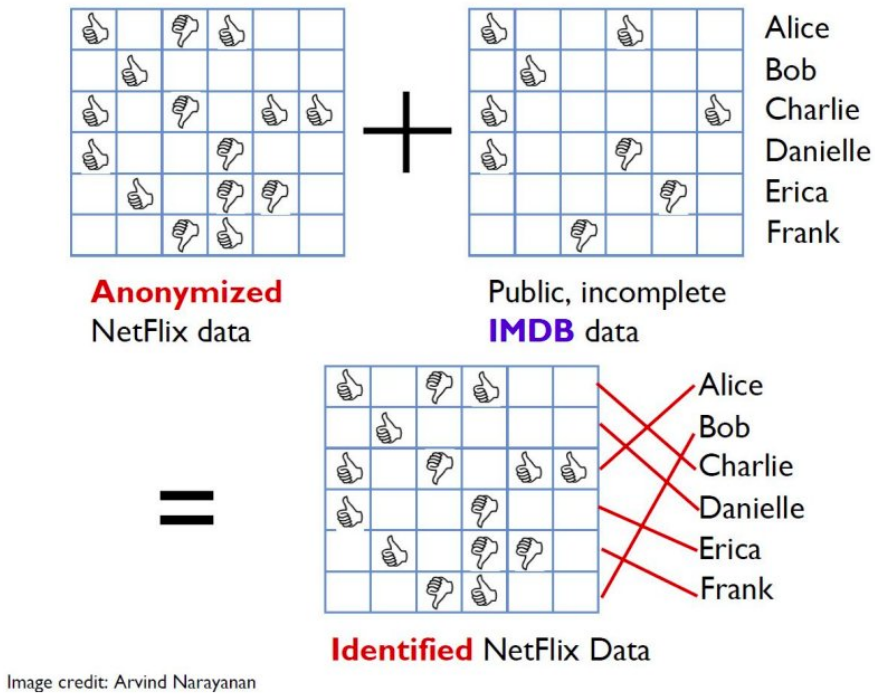

Figure 1: Figure due to Arvind Narayanan, illustrating the attack in [NS08].

was associated with either the user's name or an online pseudonym. It turns out this approach was sufficient to re-identify many users from only a few weak matches, thus giving information on these users' movie watching history, which they chose not to reveal publicly. This discovery led to a class action lawsuit being filed against Netflix, and the cancellation of a sequel competition.

Once again, this example shows that de-anonymization is insufficient to guarantee privacy, especially in the presence of side-information.

Memorization in Neural Networks

So far, we've been focused on cases which output an entire dataset, which seem to be challenging to appropriately privatize. What if we instead are releasing some function or model of the dataset? As it only gives a restricted view of the dataset, perhaps this prevents it from revealing private information? Unfortunately, once again this is not the case.

We discuss an investigation of Carlini et al. [CLE+19]. Consider training a neural network model on a text corpus Y (potentially containing sensitive information), and creating a generative sequence model f_θ. Given a sequence $x_1, \ldots, x_n$, the model is able to compute

$$P_\theta(x_1, \ldots, x_n) = -\log_2 \Pr(x_1, \ldots, x_n | f_\theta) = \sum_{i=1}^{n} \left(-\log_2 \Pr(x_i | f_\theta(x_1, \ldots, x_{i-1}))\right).$$

This quantity P_θ is known as the *log-perplexity* of the sequence. By inspecting the expression, it can be seen that a low perplexity indicates that the sequence is assigned a high probability by the

model, and a high perplexity indicates that the sequence is assigned a low probability by the model. For instance, a well-trained language model is likely to assign a low perplexity score to a phrase like "Mary had a little lamb," but a high perplexity score to "correct horse battery staple."

The question is, what if "correct horse battery stapler" *were* in the training data? Would this lead to it having a low perplexity, thus signalling that this is the case? You might think that this is not a big deal (unless this is someone's password) – but what if instead, the phrase "my social security number is 078-05-1120" were assigned a low perplexity? This might reveal the SSN of an individual in the training data. There are two core questions here:

1. Do neural networks "memorize" "secrets" in the training data?
2. If so, is it possible to efficiently discover these secrets?

Carlini et al. [CLE+19] investigate these questions with the following experimental setup. They add a "canary"[1] to the training data, which is a sensitive phrase of a particular format – we will use the example "my social security number is 078-05-1120". The question is whether this inclusion significantly lowers the perplexity in comparison to other semantically similar phrases, such as "my social security number is 867-53-0900." If so, that is an indication that the particular canary has been memorized, and may be extractable from the final model. Note that on large datasets, the canary may have to be added many times before the perplexity becomes low enough to be noticed.

Let us be a bit more quantitative: suppose we have a set of phrases $\mathcal{R}$. For the present example, this will be the set of all phrases of the form "my social security number is ???-??-????," where each ? is a digit. Imagine sorting all of these phrases in increasing order of log-perplexity according to some model f_θ: the *rank* of the canary is its index in this list. A random element of $\mathcal{R}$ would fall somewhere in the middle of the list. On the other hand, elements at the top of the list are the best candidate secrets – we consider these to be more "exposed." With this mindset, the *exposure* of some secret $r \in \mathcal{R}$ is $\log_2 |\mathcal{R}| - \log_2 \operatorname{rank}(r)$. This value ranges from 0 to $\log_2 |\mathcal{R}|$, where a large exposure corresponds to the secret being more noticeable. In particular, an exposure of $\log_2 |\mathcal{R}|$ indicates that the secret would have the lowest log-perplexity of phrases in this list.

On the bright side, it seems like extremely large models are not prone to exposing secrets in this sense. In Figure 2, the results of an experiment on Google's Smart Compose are displayed. Smart Compose is an automatic sentence completion algorithm employed in Gmail, and is trained on billions of word sequences. Even when the canary is inserted thousands of times, the exposure remains comparatively low – $|R|$ used in this experiment is on the order of 10^{12}, so an exposure of > 40 is needed for extraction, whereas it only reaches values of 10 after $10,000$ insertions. On the other hand, Figure 3 illustrates a much smaller example, with a training dataset of size roughly $100,000$, and using the social security number example given above. As we can see, with only four insertions of the canary, its exposure passes $\log_2 10^9$, at which point it can easily be extracted.

In the paper, the authors also give efficient methods of extracting exposed secrets. Naïvely, one would have to try all possible sets of phrases and determine their perplexity. A more efficient approach is possible using a Dijkstra's algorithm style method.

Finally, the authors discuss how to mitigate such memorization and exposure. Interestingly, standard techniques to avoid overfitting in machine learning are ineffective, including dropout and regularization. Differential privacy appears to be the only effective approach.

[1] A reference to the concept of a canary in a coal mine.

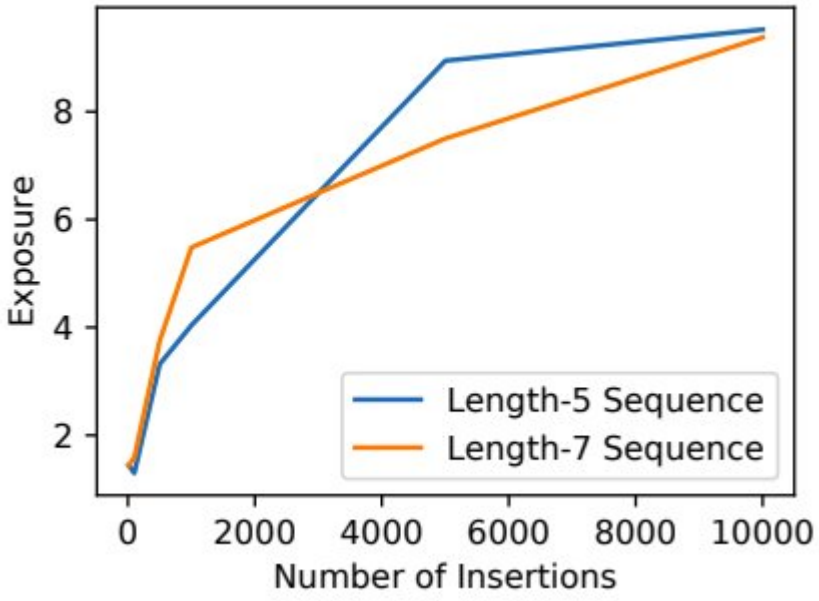

Figure 2: Figure from [CLE+19]. Even with 10000 insertions, the canary has relatively low exposure in the Google Smart Compose model.

Genomic Studies

A common setting for statistical data analysis is medical or genomic studies. While these are clearly important for scientific progress, equally important is the need for data privacy. Indeed, suppose there was a study involving only individuals who were HIV positive. If one could identify individuals who participated in this study, then this would correspond to a gross violation of their privacy. Troublingly, as Homer et al. demonstrated [HSR+08], under certain conditions, one could determine whether or not an individual was present in a mixture of DNA samples based on certain aggregate statistics. While others in the community argued that these conditions were not generally met in practice [BRS+09], nonetheless, this finding was taken quite seriously. The National Institutes of Health (NIH) in the US immediately removed several summary statistics which were previously open-source, including minor allele frequencies, chi-squared statistics, and p-values. Access to this data is now subject to an approval process, acting as a barrier to open science. Thus, privacy-respecting analyses would be a significant boon in enabling progress in this area, specifically for problems such as statistical hypothesis testing.

Massachusetts Group Insurance Commission

At some point in the mid-1990's, the Massachusetts Group Insurance Commission began a program where researchers could request hospital visit records for every state employee, at no cost. Naturally, this information is *highly* sensitive, so the dataset was anonymized. Massachusetts governor at the time (and 2020 Republican presidential candidate) William Weld promised that patient privacy was protected. If you've been paying attention, you might see where this is going.

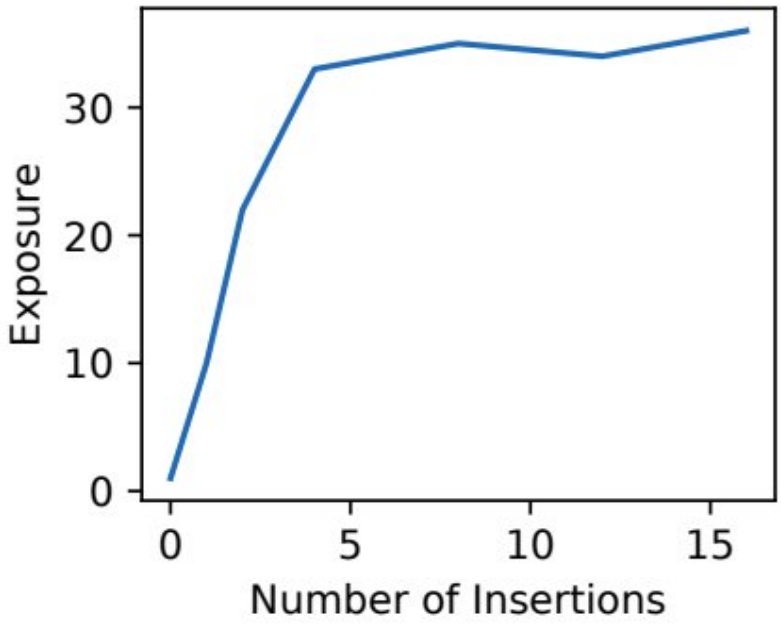

Figure 3: Figure from [CLE+19]. In a smaller example, the canary has very high exposure (and is recoverable) with only a few insertions.

Specifically, while the original dataset included information such as an individual's name, SSN, ZIP code, date of birth, sex, and condition, this was anonymized by removing identifying features such as the individual's name and SSN. Latanya Sweeney, then a graduate student in Computer Science, bought the voter rolls from the city of Cambridge, which were then available for $20. These records contained every registered voter's name, address, ZIP code, date of birth, and sex. It turns out that 87% of the United States is uniquely identified based on their ZIP code, sex, and date of birth. Thus, by cross-referencing these two datasets, it is easy to perform large-scale reidentification of individuals in the hospital visit dataset. Sweeney made a point about this, by sending Governor Weld his own medical records.

k-anonymity

In order to mitigate issues such as the above, Samarati and Sweeney introduced a notion of data privacy, known as k-anonymity [SS98]. Suppose that each point in the dataset has a set of different features. Some of these are identifiers, such as name and SSN. These would be removed from the dataset entirely. Other features are non-sensitive *pseudo-identifiers*: these are the features which *could* be associated with a person's identity. In the above example, these would include date of birth, ZIP code, and sex. Finally, there are the sensitive features, such as the condition – these should remain in the dataset, as they are they information we would like to communicate. A dataset is *k-anonymous* if, for any setting of the pseudo-identifiers, there are at least $k-1$ other points with the same settings of the pseudo-identifiers. Examples are given in Figure 4, where the left table is 4-anonymous and the right table is 6-anonymous.

	Non-Sensitive			Sensitive
	Zip code	Age	Nationality	Condition
1	130**	<30	*	AIDS
2	130**	<30	*	Heart Disease
3	130**	<30	*	Viral Infection
4	130**	<30	*	Viral Infection
5	130**	≥40	*	Cancer
6	130**	≥40	*	Heart Disease
7	130**	≥40	*	Viral Infection
8	130**	≥40	*	Viral Infection
9	130**	3*	*	Cancer
10	130**	3*	*	Cancer
11	130**	3*	*	Cancer
12	130**	3*	*	Cancer

	Non-Sensitive			Sensitive
	Zip code	Age	Nationality	Condition
1	130**	<35	*	AIDS
2	130**	<35	*	Tuberculosis
3	130**	<35	*	Flu
4	130**	<35	*	Tuberculosis
5	130**	<35	*	Cancer
6	130**	<35	*	Cancer
7	130**	≥35	*	Cancer
8	130**	≥35	*	Cancer
9	130**	≥35	*	Cancer
10	130**	≥35	*	Tuberculosis
11	130**	≥35	*	Viral Infection
12	130**	≥35	*	Viral Infection

Figure 4: Figure from [GKS08]. The table on the left is 4-anonymous, the table on the right is 6-anonymous.

While this resolves some issues raised in the Massachusetts GIC example above, this is still vulnerable to several attacks we have discussed before. For instance, suppose we know that our 35-year old friend visited the hospital corresponding to the left table in Figure 4. Then we would be able to conclude that they have cancer. Alternatively, suppose we know someone who is 28 years old visited the hospitals corresponding to both tables: we can infer that they have AIDS. Such issues were highlighted by Ganta, Kasiviswanathan, and Smith [GKS08].

References

[BRS+09] Rosemary Braun, William Rowe, Carl Schaefer, Jinghui Zhang, and Kenneth Buetow. Needles in the haystack: identifying individuals present in pooled genomic data. *PLoS Genetics*, 5(10):1–8, 2009.

[CLE+19] Nicholas Carlini, Chang Liu, Úlfar Erlingsson, Jernej Kos, and Dawn Song. The secret sharer: Evaluating and testing unintended memorization in neural networks. In *28th USENIX Security Symposium*, USENIX Security '19, pages 267–284. USENIX Association, 2019.

[GKS08] Srivatsava Ranjit Ganta, Shiva Prasad Kasiviswanathan, and Adam Smith. Composition attacks and auxiliary information in data privacy. In *Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '08, pages 265–273, New York, NY, USA, 2008. ACM.

[Hal14] Jason Hall. https://www.reddit.com/r/bigquery/comments/28ialf/173_million_2013_nyc_taxi_rides_shared_on_bigquery/cicr3n2/, June 2014.

[HSR+08] Nils Homer, Szabolcs Szelinger, Margot Redman, David Duggan, Waibhav Tembe, Jill Muehling, John V. Pearson, Dietrich A. Stephan, Stanley F. Nelson, and David W. Craig. Resolving individuals contributing trace amounts of DNA to highly complex mixtures using high-density SNP genotyping microarrays. *PLoS Genetics*, 4(8):1–9, 2008.

[NS08] Arvind Narayanan and Vitaly Shmatikov. Robust de-anonymization of large sparse datasets. In *Proceedings of the 29th IEEE Symposium on Security and Privacy*, SP '08, pages 111–125, Washington, DC, USA, 2008. IEEE Computer Society.

[Pan14] Vijay Pandurangan. On taxis and rainbows: Lessons from nyc's improperly anonymized taxi logs. *Medium*, 30, 2014.

[SS98] Pierangela Samarati and Latanya Sweeney. Generalizing data to provide anonymity when disclosing information. In *Proceedings of the 20th ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems*, PODS '98, page 188, New York, NY, USA, 1998. ACM.

[Who14] Chris Whong. Foiling nyc's taxi trip data. *Chris Whong*, 2014.