CCS 2019 Talk Summaries

Sebastian Meiser, Visa Research

November 12, 2019

This document presents a few summaries of talks I attended at CCS 2019. I don't guarantee that the summaries are comprehensive or that they capture all subtleties of the presented matter; if you find any technical inaccuracies, please contact me about them.

1 Pre-Conference Workshop: TPDP

1.1 Encode, Shuffle, and Analyze (ESA) Revisited: Strong Privacy despite High-Epsilon

Speaker: Abhradeep Guha Thakurta, Google Research, UC Santa Cruz

Assume we have a number of devices use an anonymizer and local DP to achieve some central differential privacy. The anonymizer can either use summation or shuffling to achieve this goal. Naturally we focus on the second part here.

What we do is we take the locally DP objects, remove identifiers (if there are any), shuffle them, and release them. We want to start with weak local DP, i.e., with a high epsilon ($\varepsilon > 1$) and still achieve strong central differential privacy: We get a boost of about $\frac{1}{\sqrt{n}}$ for ε.

To this end, we look at three ideas:

- **Attribute fragmenting:** We split one-hot vectors into the separate bits, then shuffle them, and get some utility in $\Theta\left(\sqrt{\frac{\log k}{ne^{\varepsilon_{\text{local}}}}}\right)$. Note that if we have t records and a local $\varepsilon_{\text{local}} = 1$ we get local DP of $t \cdot \varepsilon_{\text{local}}$.
- **Record fragmenting:** I'm not quite sure what exactly happened here; I think the result is that instead of an ok central DP ($\varepsilon = 1.5$) trade-off that comes with a horrible local DP guarantee ($eps \approx 25$), we can have a local DP of $\varepsilon = 1$ and still get central DP with $\varepsilon = 1.5$. This degrades the utility, but Abhradeep assures us that it's not that bad.
- **Crowds:** We can group records to achieve a better local DP / utility trade-off. We split our data into crowds and analyze them. A cute idea here is to add Laplace noise $\text{Lap}\left(\frac{1}{\varepsilon_{\text{shuffle}}}\right)$ (and subtract a large enough

constant) to the count of records it has and then drop as many records as required to meet the count. If we still come up with a number higher than the actual count, we have a distinguishing event.

1.2 DPella

Speaker: Elisabet Lobo Vesga

We know how to do queries with DP and how to estimate the accuracy. However, what happens to our accuracy if we want to add and combine the results of several queries?

In comes DPella, a Haskell library, which allows us to keep track the privacy and accuracy of the (combined) queries we ask and to find out the privacy budget used by a program (via symbolic execution). Similarly, we can get an estimate of the accuracy of the program. That sounds pretty interesting.

So far, they only consider the Laplace mechanism, but they are working on integrating the Gauss mechanism as well.

1.3 Private Stochastic Convex Optimization with Optimal Rate

Speaker: Abhradeep Guha Thakurta, Google Research, UC Santa Cruz

When we design a (differentially private) learning algorithm, we have to consider population risk. Here we have a convex loss function, which makes things easier in many ways. The excess population risk is the expected loss on a random sample, when compared with the minimal such loss. $\max\left(\frac{1}{\sqrt{n}}, \frac{\sqrt{d}}{\varepsilon n}\right)$, the first part of which is what we get for the non-private case and it turns out we often still achieve that.

We look at a noisy SGD with a batch size of approximately $\max\left(\sqrt{n}, \sqrt{d}\right)$; the number of rounds we need is about $\min\left(n, \frac{n^2}{d}\right)$.

1.4 Lessons learned from the NIST DP Synthetic Data Competition

Speaker: Ryan McKenna, University of Massachusetts, Amherst

The talk is about differentially private synthetic data, which is very interesting for a number of reasons. If we have private synthetic data, we can use arbitrary mechanisms on them and we can use it however we want without needing to keep track of any budget.

The speaker discusses the competition he partook in. The data was US census data with 98 attributes and 661k individuals. All the data fields was made of integers between 0 and a known maximum. The synthetic data was judged on: accuracy on all 3-way marginals (10^13 queries) and an approximation of high-order conjunctions (such as "how many records have an age in range X and income in range Y?").

The way they approached this was to first compute a range of noisy aggregate statistics and to then use an inference engine to create synthetic data from it.

1. Measure 1-way marginals using the Gaussian mechanism, treat counts below a threshold as zero.

2. Construct a correlation graph between any pairs of attributes. Each attribute is a node in the graph and the edge weights correspond to the correlation between the attributes. They construct this on the provisional dataset that is assumed to be public (we need to sacrifice the privacy of this dataset). We find a maximum spanning tree of the graph (of the provisional data), then measure 2-way marginals in a differentially private way on the actual data for each part of the tree.

The top 4 solutions of the competition were fairly similar, with the winning one (the one presented) being unique in terms of the inference engine used. Their mechanism ran in 30 minutes.

1.5 Full Convergence of the iterative Bayesian update and applications to local differential privacy

Speaker: Catuscia Palamidessi

In the local DP setting, every user can theoretically set their own privacy level and they don't need to be the same over all users. In this talk we focus on the statistical utility of the data and we try to retrieve the original distribution of the data.

Catuscia presents the iterative Bayesian update: they start with any distribution with full support, e.g., uniform, and then update the distribution iteratively. This approach achieves a pretty good approximation of the original distribution; the work presented fixes a bug of this approach, extends it to more mechanisms, and compares the technique to other inversion techniques.

1.6 Differentially private real summation with single- and multi-message shuffling

Speaker: James Bell

This work too talks about the shuffle model, here using the randomized response mechanism. In their analysis, they allow the adversary to know whether each participant (other than the one of interest) lied or told the truth. Each participant that lied naturally introduces some noise that helps protect the one remaining real entry. Thus, this technique relies on at least a certain number of other participants to add noise correctly in order to achieve privacy.

The work(s) also look at reducing the amount of communication necessary to achieve a good trade-off when more messages per party are allowed. They reduce the number of required messages from about $\sqrt{n}$ to $\log(n)$.

1.7 Differentially private release of synthetic graphs

Speaker: Marek Elias

We look at social networks and Marek starts off the talk with the example of twitter, more precisely, re-tweets between hardcore democrats and republicans.

To achieve privacy, we want to start with a graph G and create a differentially private graph G' and we want to preserve the weight of cuts. Existing work by Gupta et al. creates a fully connected graph with Laplace noise on the edges. That's okay if we have a graph with lots of edges already, but for the sparse graphs we're looking at, it would actually be more correct to output an empty graph.

Here we use a sparsifier that preserves cut sizes with a small multiplicative error, however, exponential time is required. This work provides the first non-trivial guarantee that can be computed in polynomial time. I'm not sure how good / useful this actually is in practice.

1.8 Privacy hypothesis testing via robustness

Speaker: Audra McMillan

Audra focuses on how to design private testing algorithms. We start with yes-or-no questions and generate hypotheses out of them.

Here, a test gets a database as an input distinguishes between the null hypothesis and an alternative hypothesis, i.e., it outputs a bit. Differential privacy is useful here not just for privacy, because it provides stability and that's a valuable property to have anyway.

We look at two different problems here:

- in simple hypothesis testing we try to compare two different distributions. That's very much standard DP stuff.

  In a non-private way, we solve this problem as follows: given a dataset X, we output P if it's more likely than Q and Q otherwise.

  We can rewrite this as follows:

  $$L(X) = \sum_{x \in X} \log \frac{P(x)}{Q(x)}$$

  Now we can replace the test by:

  $$LLR(X) = P \text{ if } L(X) \geq 0, Q \text{ if } L(x) < 0$$

  This is pretty similar to an operation on the privacy loss. We seem to get some privacy for free here by making the tests more robust.

- Alternatively we can look at identity testing in high dimensions. Here, we basically compare a uniform distribution from a product distribution that's far from the uniform distribution. Here a main insight is that the

global sensitivity is extremely large with a high number of dimensions. However, the worst-cases that actually give us such worst-cases are really rare and don't at all look uniform, so we should be able to distinguish them easier anyway.

They define a set of "good datasets" coming with a test that is insensitive and that rejects datasets that aren't in the set of "good datasets". As a first step, this test is used to reject things that are obviously non-uniform.

Fortunately there is a function $\hat{T}$ that has a sensitivity of T on the good dataset and that satisfies $\hat{T}(X) = T(X)$ in that region. Basically, $\hat{T}$ is like the test we want to use, but insensitive anywhere. We then use $\hat{T}(X) + \text{Lap}\left(\frac{\lambda}{\varepsilon}\right)$

If we sample from the uniform distribution then any two samples should be independent, i.e., their inner products should be small. Consequently we define the good region as datasets with small inner products and that don't have a bias in an individual dimension that is too high.

1.9 Private hypothesis selection

Speaker: Mark Bun

Given a publicly known collection of distributions H and some i.i.d. samples $x_1, \ldots, x_n$ from some unknown distribution P and we want to find a hypothesis $h \in H$ that is close to P in total variation distance: If there is a hypothesis $h^* \in H$ s.t. $\text{TV}(P, h^*) \leq \alpha$ then with high probability we output some $h \in H$ with $\text{TV}(P, h) \leq O(\alpha)$.

This work provides a robust variant of hypothesis testing, gives sample-efficient algorithms fir distribution learning and can be used as pre-processing for other applications.

Why use total variation distance? It's mathematically convenient and it's insensitive to low probability events.

In a non-private way we can achieve this with a Scheffe Tournament that lets the hypotheses compete against each other. If one hypothesis h_1 wins against another $h2$, then the winner is at most some small error term away from the $\min\{TV(h_1, P), TV(h_2, P)\}$.

To achieve differential privacy, we add noise to the tests (here, Laplace noise). If the number of samples is high enough, we can achieve privacy somewhat easily, but we'd like to avoid this large number of samples.

This work uses the exponential mechanism and encodes something smart as the quality function q of the exponential mechanism. We can't use the number of contests won, as the sensitivity of that would be really high. Mark introduces a slightly modified contest that introduces draws to the picture and then sets q as the minimum number of samples that have to be changed for h to lose at least one contest.

1.10 The search for anonymous data: attacks against privacy preserving methods and systems

Speaker: Yves-Alexandre de Montjoye

Yves starts his talk by making clear that this is an interactive session. The first question he asks is how we can find a balance between data (which is very useful and enables great utility, e.g., via machine learning) and privacy.

In practice, people try to preserve privacy by providing anonymity of the data, which fits well to both the GDPR (in Europe) and the CCPA (in California). Yves highlights that this is problematic, since large datasets of released anonymized information were de-anonymized later on by researchers.

First Act: Pseudonymization Yves holds the opinion that mere pseudonymization does is not sufficient to provide privacy. From a privacy-perspective, that is certainly true.

What does it take to identify someone in a dataset? As one example, he shows that in a mobile phone dataset 4 points of hour + location are enough to uniquely identify 95% of the users.

Second Act: De-identification Here, we try to make it harder for the attacker to find data points, e.g., by adding noise. For the mobile phone dataset, they added a significant amount of noise to every data point (up to several kilometers). We have that the uncertainty $\sim (v \cdot h)^{\frac{-p}{100}}$, where h is the soatial resolution, v the temporal resolution and p the points known to the attacker.

Yves then discusses the argument "you can never be sure" (that the person you think you found even was in the dataset). The question here is, whether or not we can quantify this uncertainty. Basically: how likely are we to have found the actual record?

To answer this question, they took a small subset of the dataset (way less than 1%), learned to re-identify records and then test whether they can predict if a record s=is unique or not, which they evaluate on the whole dataset. [1]

Yves confludes this acy by saying that anonymization (in the traditional, de-identification sense) doesn't work for high dimensional data.

Diffix, a heuristic query-based system Diffix is a privacy preserving system, (although it doesn't provide differential privacy). The way it works is that every query is answered with a combination of static and dynamic noise. For every condition of the query (say: the number of people with age=40, dept=Computing, high-salary=True), they have a static noise term and they also add a dynamic noise term each. They hoped to achieve (virtually) unlimited queries with somewhat little noise.

Yves and his group developed an attack on this system.

The assumptions are that there is only one target and the attacker wants to infer a specific attribute of Bob (binary).

[1] their model is available here: `cgp.doc.ic.ac.uk/individual-risk`

They send two queries, one with one less condition. Now the static noise terms of the same conditions remain the same and only one noise term (of the condition that was changed) as well as the dynamic noise terms remain. Now, if Bob, say, has a high-salary and if they can assume that Bob is also unique in these attributes, they get two different distributions. They then exploit the background knowledge of the attacker, use a simple likelihood-test and can then infer Bob's attribute with high certainty, depending on the number of known attributes of Bob (and on the dataset); for most datasets, the attacker needs background knowledge about between 10 and 30 attributes.

Diffix proposed a patch that seems to mostly target the specific attack (by disallowing "dangerous queries"), but that does not seem to fix the underlying issue.

2 CCS Main Conference, Tuesday

2.1 Privacy I – Watching you watch: the tracking ecosystem of over-the-top tv streaming devices

Speaker: Ben Burgess, Princeton University

When moving from cable TV to streaming services, one of the cheapest options is to use a dedicated OTT streaming device. These devices introduce new privacy concerns. Many of these devices, such as the one from Roku and Amazon's Fire TV come with an app-store of sorts. Many channels make money via ads and the vast majority of them implement trackers. The platforms themselves seem to disregard privacy policies.

Analyzing the devices is not easy, since they encrypt their traffic (and thus the traffic cannot be easily analyzed). This work analyzes the OTT devices via man-in-the-middle attacks; they intercept all HTTP streams and try to intercept HTTPS streams via self-signed certificates. Ben mentions that "some apps validate certificates".[2]

They rooted the Amazon Fire TV device to get more access and used Frida to bypass channel-level certificate pinning.

To automate their crawler, they recorded the keystrokes (of the remote) required to get a video to start. They then used the most common ones to try to start videos automatically with their crawler. They used audio detection to verify that a video is played and fast-forward to maximize the number of ads they see, as the ads are what actually leads to tracking.

Results They used available tracking lists to identify tracking domains (such as doubleclick). To get more domains they listed the domains that were sent the ad ID and were contacted by more than one channel, which allowed them to identify lesser known tracking domains.

[2]Comment: that's a statement I find concerning for different reasons: it seems that many channels poorly implement their certificate validation.

They saw that in addition to the ad ID, a variety of other information was sent to the ad trackers: This included the video title for many trackers!

While Roku's privacy option removed the ad ID, but did still send the serial number of the device, Amazon's system still sent some ad ID's and barely reduced the other information sent.[3] Ben concludes that the platform's privacy options have a minimal effect and platform independent options are difficult to use in this system.

2.2 Privacy I – Oh, the places you've been! User reactions to longitudinal transparency about third-party web tracking and inferencing

Speaker: Miranda Wei, University of Chicago

This very colorful talk is about understanding web tracking. Browser defenses show the number of trackers, but not necessarily who is tracking you and whether they have seen you before.

Privacy dashboards give aggregate information on their inferences, but doesn't tell you what specific information they have collected exactly. Google ads and Facebook, as examples, have hundreds or thousands of pieces of information that allows them to draw fine-grained inferences.

This work is a longitudinal study about which trackers made which inferences about users based on which browsing activity. None of the existing solutions provide this information directly, so they designed their own browser extension, called *tracking transparency* for their study. It works as follows:

- The extension uses a topic modeling algorithm using about 2000 Google ads categories then searched these categories in Wikipedia, extracted and pre-processed text and then compared this text with what the user typed.
- Inferences were made locally and stored locally
- The extension collects meta-data of the websites used and keeps track of the trackers encountered.
- The users can see the inferences:
  - First, the user is shown sites visited, trackers encountered, etc.
  - the more detailed view presents what interests that could have been inferred about the user (the user can toggle between "last 24 hrs", "last week" or "all time"; can consider which popular or less popular topics could have been inferred, etc.).
  - The user is shown how many trackers (and which, and how many times) could have made a specific inference.

[3] Caveat: In the questions, Ben states that they didn't check whether activating the privacy setting reduced information such as the video title.

– Users can also see which trackers and how many they have seen and can look at how often they have been encountered and which interests they could have inferred.

They started off with a usability study with 13 to improve the interface of their extension and then ran a user study with 425 participants that used the extension for one week. They had 6 study conditions that differed in what information they presented the participants.

Results The participants cumulatively visited more than 1 million web pages. about 40% of them used ad- or tracker-blocking tools. The information shown to the participants generally improved their awareness of tracking and their knowledge of tracking.

Participants adjusted upwards their estimate about how many trackers will track their behavior (although still under-estimating them), and stated that they'd be more likely to use tracker-blocking tools in the future. Both effects were stronger with more data shown to them. [4]

2.3 ML Security I – Neural Network Inversion in Adversarial Setting vie Background Knowledge Alignment

Speaker: Ziqi Yang, National University of Singapore

This works aims at inverting neural networks, similar to other *model inversion attacks*; they aim to reconstruct images from the prediction scores of a neural network. In the adversarial setting, the adversary has no access to the training data. Moreover, Ziqi only allows their attacker to have partial access to the prediction scores.

Traditional training-based inversion methods allow the inverting model to be trained on the original data. This work instead requires the adversary to come up with their own data. Here, too, training-based inversion is performed. The model takes the prediction scores as inputs and is trained to predict a face image. The adversary here just crawls the internet for face images and then uses them by feeding them into the classifier; the outputs are truncated, fed into the inversion model that now tries to recreate the image.[5]

What stands out from their results is that their approach can produce recognizable faces on slightly truncated prediction scores. When using less than 100 (out of 530) features, their approach naturally doesn't reconstruct very recognizable faces anymore.

[4] Overall, their tool seems very helpful to provide insights into the tracking landscape and could be used for educational purposes; the extension is available here: `git.io/trackingtransparency`

[5] Comment (privacy view): this is an interesting paper and seeing how well the partial prediction scores can be used to create a fairly similar face is definitely impressive. This attack doesn't break privacy though, it just makes it clear that we need to consider prediction scores to be as sensitive as the actual inputs to the neural network.

Extension: white-box setting The adversary here can jointly train the classifier and the inversion model. They compare this extension with their black-box approach (from above); they find that this doesn't improve over their black-box inversion much. Moreover, if they enhance the auxiliary information used by the black-box approach, they have comparable results as the white-box inversion.

2.4 ML Security I – Privacy risks of securing machine learning models against adversarial examples

Speaker: Liwei Song, Princeton University

Liwei defines what he considers trustworthy machine learning: it should be *privacy preserving*, i.e., resistant against membership inference attacks, and *robust*, i.e., resistant against adversarial attacks which cause misclassification during test-time. [6]

This works presents new membership inference attacks (MIA) against robust models and actually claims that increasing robustness makes the model more vulnerable to MIA. The intuition here is that defense methods against adversarial examples artificially change the decision boundaries and this can depend more on the individual samples in the training data.

The defenses can be categorized into:

- empirical defenses that add slightly modified samples to the training procedure.
- verifiable defenses where the loss is computed over the whole area around each training sample.

To validate their intuition, Liwei tries to quantify the influence of training data on the model:

1. they randomly choose and remove a single training sample,
2. they retrain the model without that,
3. they compare the difference of the selected sample's prediction confidence of both models.

A second way of verifying this intuition is to check whether there is a larger divergence between model predictions on training data and test data with robust training. They empirically validate this and, indeed, the robust network has a much more pronounced difference in cross-entropy loss when comparing a robustly trained model with a "naturally trained" model.

Their MIA basically only checks whether the prediction confidence on either a regular or an adversarially modified input is above a threshold. If it is, they

[6] They consider evasion attacks that try to make models misclassify during test time, not poisoning during training time.

consider the sample to be in the training data. If the confidence is lower, they consider it to be not in the training data.

On benign points, their attack seems to work a little bit on the naturally trained model (57%) and much better on the robustly trained model (75%). Unsurprisingly, the difference is much more extreme on adversarially modified points, where the naturally trained model will always have a low confidence, but the robust model will still have confidence. The accuracy of their adversarial classifier on robust models is not significantly higher.

As the perturbation budget (for robustness) increases, the accuracy of their attack also increases. Finally, they compare their attacks on models trained with other robustness techniques from the literature and again, the accuracy of their attacks is higher than against the "naturally trained" model. The verifiable defenses provided the most pronounced increase in attack accuracy. [7]

2.5 ML Security I – MemGuard: Defending against black-box membership inference attacks via adversarial examples

Speaker: Jinyuan Jia, Duke University (recorded)

Private attributes can often be inferred from public information. As an example, Cambridge Analytica used public Facebook likes and known attributes (from users that disclosed them) to train a model that could then infer these attributes for users that didn't disclose them (from their public Facebook likes).

Jinyuan's idea is to use adversarial examples to attack attackers, hoping they'd make wrong inferences. While existing works have looked at parts of this problem, they have not yet considered membership inference attacks and that is what this work looks at.

Their approach, MemGuard, noises the prediction scores output by the machine learning classifier they want to protect. Ideally, this noise confuses the attack classifier for the MIA. As the defender doesn't know what exactly the attacker will do, the defender trains their own MIA classifier. Since the decision boundaries are probably similar for the attacker's classifier (transferrability), the defender then uses their own MIA classifier to choose the noise. [8]

In their evaluation they also consider the *label loss*, i.e., the loss in accuracy of their prediction, which they defined as 0 if the prediction does not change and as 1, if the prediction does change.

[7] In the questions section, Liwei is asked whether his results can be explained with overfitting. He admits that, indeed, overfitting is slightly increased by robustness. He explains that this effect is insufficient to explain the increased vulnerability, since overfitting (as defined by ML people) is not very pronounced. ML people consider a model to be overfitted if and only if the test accuracy (i.e., the amount of generalization) is reduced. As a privacy person, I naturally care less about the test accuracy and would consider a model to overfit if it unnecessarily memorizes the training samples and this is exactly what seems to be happening in these robustly trained models.

[8] Comment: I really wonder how robust their defense is. Can't the attacker use different metrics and/or train the MIA classifier with robustness?

It seems their confidence score distortion budget needs to be somewhat significant for MemGuard to be effective; however, when compared to other defenses, it appears MemGuard is much stronger, particularly when measured by the label loss.

2.6 ML Security I – Procedural noise adversarial examples for black-box attacks on deep convolutional networks

Speaker: Kenneth Co, Imperial College London

Kenneth talks about evasion attacks and how the literature has mostly focused on input-specific noise. He explains that universal adversarial perturbations, i.e., perturbations that work for any image, work very well too. Such patterns can be found via machine learning, but Kenneth is more interested in alternative methods for generating them. In comes procedural noise stemming from procedural functions close to what has been used for a variety of other use cases. Examples include Perlin noise and Gabor noise, both of which have only 4 parameters, which is significanly less than the hundreds of thousands of parameters for image datasets, i.e., the search space is significantly smaller here.

The attacker here works as follows: they can choose pertubations that are added to images, fed into a classifier (with a structure unknown to the adversary) and the adversary is only given the successes and failures in terms of classification. Even if the noise parameters are chosen completely at random, they already have a really high evasion rate (of about 80% for noise of 12%).

They also look at input specific evasion, i.e., they try to find parameters for each input image so that the image is misclassified. For Perlin noise they found that for several training sets for about or more than 95% of images there was some procedural noise than led to misclassification.

They use Bayesian optimization to find a more efficient way to find good parameters. Now the attacker performs input-specific perturbations. When comparing their results to previous work, it seems that Bayesian optimization is much more query-efficient to provide a very strong, comparable success rate (of $> 90\%$).

Finally, they show that their noise generalizes to other visual tasks other than image classification. They looked at object detection models and here, too, the procedural noise greatly damages the recall by either masking objects or flipping labels. Since it doesn't seem to introduce new objects, they conclude that the noise masks existing objects.

Interestingly, when visualizing the early layers of the models, the activations of these layers look very similar to Gabor noise. Since the early layers are often used as a feature extractor, the procedural noise's impact on these layers is particularly troubling (as the same early layers might be used in many applications). [9]

[9] Their procedural noise attack is available online at:
`github.com/kenny-co/procedural-advml`

In the questions and answers, Kenneth is asked whether this technique can be used for targeted attacks; there also is the (slight) criticism that the noisy images are visually slightly similar to the (wrong) labels predicted by the models, e.g., shower curtains or corals with wavy patterns.

2.7 Privacy II – Analyzing subgraph statistics from extended local views with decentralized differential privacy

Speaker: Haipei Sun, Stevens Institute of Technology

Decentralized social network analysis has been performed with local differential privacy, but Haipei isn't impressed with the utility/privacy trade-off.

The local view of a user consists of their 1-hop friends. An extended local view additionally includes 2-hop friends. As users can see their own extended local view, we can perform analyses by asking the user higher level questions, such as "how many friend triangles are you in?" (A is friends with B and C and B and C are also friends), instead of lower level questions that might require access to the friend list. Such a question, however, can over-report connections (as the same edge may occur in two different triangles). This is an issue for computing the sensitivity of each edge.

The work focuses on edge-level differential privacy and introduces a new definition which they call decentralized differential privacy (DDP) for edges. They feel the need for this, since local DP assumes that each user only holds their own data. [10]

Definition of DDP [I didn't catch the definition; might add it later]

They use a 2-phase approach:

1. They apply a DDP algorithm to get the local sensitivity. Users report the noise degree they'd like to use to the data collector.

2. The data collector uses the second largest noise degree λ and sends that to each user.[11]

3. The users report noisy triangle counts.

As a variant, they sort the noisy degrees and only ask a number of h users that reported the highest values to report some more noisy values. These values are then used to choose the level of noise.[12]

[10] That's not exactly true, but local DP can be used like that.

[11] Comment: I presume that they somehow hope that the "local" sensitivity reported in this way is still much smaller than the global sensitivity. Also, probably they put up with a small δ event in case they should have used the largest one.

[12] Comment: Edge level privacy is extremely weak. I' not sure what to make of this. Also, choosing the noise in such a data-dependent way requires careful accounting.

2.8 Privacy II – How to accurately and privately identify anomalies

Speaker: Hafiz Asif, Rutgers University

We want to find anomalies (such as fraudulent transactions) while protecting privacy. Anomalies are often defined on a data-dependent level and privacy tries to protect every data point, including outliers.

Hafiz proclaims that differential privacy is not a good choice here. Thus, he wants to use a different privacy definition, which he calls sensitive privacy. Sensitive privacy requires privacy protection for every record that is or becomes normal under a small change in the database. The idea here is that records that are outliers will robustly remain outliers if the database changes slightly (a number of k records are added or removed).

Outliers here are defined as having at most a small number of close-by data points.

Basically, sensitive privacy requires pure DP for all inputs that differ by one record that is k-sensitive with respect to x or y.[13]

[13] Comment: This is still differential privacy; the only difference is in the definition of neighboring datasets. As they want to treat outliers differently and don't care about their privacy, they choose not to protect them. This is a bit subtle due to the fact that outliers are defined necessarily in a data-dependent way. Thus, publishing the outliers outright could leak information about the "normal" data points.