CCS 2019 Talk Summaries

Sebastian Meiser, Visa Research

November 14, 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 the (cleaned) text of the current web page the user was on.
- 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. Outliers here are defined as having at most a small number of close-by data points.

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).

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]

2.9 Privacy II - Differentially private nonparametric hypothesis testing

Speaker: Adam Groce, Reed College

Related to the previous talks on differentially private hypothesis testing, Adam wants to give (ε)-differentially private versions of existing hypothesis tests. To this end, this work uses Laplace noise, scaled appropriately, to get differential privacy.

1. They compute a test statistic $t = T(D)$
2. They compute a p-value $p = Pr[T(D) \geq t|H_0]$, where H_0 is the null hypothesis.
3. If the p-value is small enough, the hypothesis is very likely.

A lot of the earlier work is missing a computation of rigorous p-values. Also, private approximations of an existing standard statistic required plenty of data points to be useful.

This work mostly looks at testing for the independence of two variables, where one is categorical (e.g., smoking status) and one is continuous (e.g., blood pressure). They modify an existing non-parametric statistic. Both this statistic and the modification have slightly worse performance in the non-private setting, but they are fairly resistant to adding noise.

[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.

3 CCS Main Conference, Wednesday

3.1 Privacy III – Five years of the right to be forgotten

Speaker: Kurt Thomas, Google

The talk is about the right to be forgotten and studies the requests users send to Google for information to be removed. A large number of statistics are presented, many of which are exactly as expected.[14]

Interestingly, about half of the websites that people wanted to have removed from searches for their name went offline after the GDPR came into effect.

Some of the metntioned statistics are: Most people that ask for removals are private individuals (84%) and much less from politicians. There are heavy hitters (the top 10k of requesters, less than 10%, request the removal of more than 34% of URLs). There is bias in terms of countries, with France being much more active than Greece.

3.2 (Un)informed Consent: Studying GDPR consent notices in the field

Speaker: Christine Utz, Ruhr-Universität Bochum

Christine starts her talk about cookie notices that pop up on websites. These started to appear after the GDPR came into effect, as consent is one of the bases listed in the GDPR. The aim of this work is to figure out how users interact with these pop-ups and how the different UIs influence the behavior of users.

To this end, they collaborated with a German e-commerce website based on Wordpress. They used a modified Wordpress plugin to test different cookie pop-ups with different parameters. They took a sample of 1000 existing cookie notices, categorized them in terms of size, position, whether or not it blocks access to the website, the text of the notice, the choices offered and nudging or dark patterns (such as highlighting the "accept all" button in green or just showing an 'OK' button while showing a non-button with settings) and potentially links to the privacy policy.

1. They experimented with the position (showing a notice with a binary choice). The positions included the 4 corners, a banner at the top and a banner at the bottom. The position had a significant impact on the interaction rates, with the highest rate of interaction when it was placed on the bottom left. They conclude that probably that's because it's more likely to block some information on the website at this position.

2. Using the "best" position, they experimented with nudging. They had different settings, where they also distinguished between opt-out and opt-in of tracking. They found that nudging had a strong influence, which really isn't too surprising. It seems that the vast majority of users simply clicked submit without changing the preferences.

[14] All results are available in Google's transparency report.

3. Finally, they tried out slightly different wording and whether or not adding a link to the policy changed things. The existence of a privacy policy link didn't have a significant effect.

In the post-interaction questionnaire, the most common reasons for clicking the notice was being annoyed by it (and wanting it to go away) and the expectation that the website wouldn't work properly unless they clicked on it.

Takeaways

- The position of a banner plays a role.
- Nudging and preselections play a big role.[15]
- Privacy links and wording weren't very influential.
- There is a widespread misconception that websites cannot be used unless cookies are accepted.

3.3 Moving beyond set-it-and-forget-it privacy settings on social media

Speaker: Mainack Mondal, University of Chicago

When we put up content on social media and choose the privacy settings for it, these settings might be appropriate at the time (e.g., putting up a silly picture while young and only having similar friends), but don't revisit these settings later, when the settings might become inappropriate (e.g., later when they picked up other friends, such as colleagues or their students).

This study measures the privacy activity and preferences of Facebook users. They collected data by creating a privacy-preserving data-collection infrastructure, then recruited 78 users. They surveyed them about Facebook usage and privacy preferences. They asked them for consent and then also presented them with the highlights of the privacy impact on them, to make sure the users understood what they were agreeing to.

Methodology They created a Facebook browser plugin that needed to be installed by participants. This plugin would send the Facebook cookies to the research team. The data processing was done automatically, no humans ever saw the raw data. Names were hashed and no images were collected. They did collect data from the activity log (all the activities the user has ever done on Facebook). They again pseudonymized the data.

In the final, post-specific survey they chose 5 random posts per user, presented the post and the current privacy settings of that post to the users, asking them to either keep the setting, delete the post or change the setting.

They asked them about 6 specific (randomly chosen?) friends per post (that were currently able to see the post), asking them whether they want to keep

[15] There is an upcoming ruling on whether preselections can be considered valid consent.

sharing the post with that other user, whether they don't care or whether they want to stop sharing the post with that user.

Results and statistics Two thirds of the participants were female. The median participant had a 10 year old Facebook account with about 1800 posts. Most old posts (and new posts) were shared with all friends. Naturally, people accumulated friends over time, which means that the old posts became accessible by more and more people.

In a further study they showed the users a number of posts. The participants would want to change the settings of about 25% of their old posts, with slightly more posts to be limited more, but also a significant number of posts to be shared more widely.

Automatic privacy assistant Finally, Mainack looked at automated classifiers and how much they could help to change these settings. Their vision would be to predict that a user might want to stop sharing a given post with a specific friend. To this end, they trained classifiers to predict which posts/friend pairs for which users would be most interested to stop sharing that specific post with that specific friend.

They compared their random forests and XGBoost predictions with a baseline of purely random suggestions and another prediction based on the number of interactions with those friends (where they would choose friends users were less likely to interact with).

They measured the precision as "How many of the suggested posts would the users actually like to stop sharing?". They found that for the top 10 recommendations, the precision is above 80%, which is promising.

3.4 Quotient: two-party secure neural network training & prediction

Speaker: Nitin Agrawal, University of Oxford

As neural networks are trained on sensitive data, it is ever more important to consider security aspects of the training phase. Novel legal frameworks provide additional motivation for looking into securing the training process. There are many works that look at secure multi-party computation (MPC) to secure the data during training. This particular work continues the line of work started by *Secure ML*, which is based on two-party MPC.

Nitin looks at the types of computations required for neural network training, such as matrix vector multiplications, activation functions (such as ReLU), batch normalization, as well as gradient computations and weight updates.

With MPC we have some limitations that we need to respect in order to have feasible protocols; these include that we don't compute square roots, have neither multiplication nor division, and that we use fixed-point arithmetics. These limitations have led to very restricted networks that, e.g., only have 2-bit weights $w \in \{-1, 0, 1\}$. Generally the idea seems to be train networks in ways

that work well with MPC techniques. As an example, Nitin presents multiplication via oblivious transfer for the matrix multiplication part and introduces a number of other little tweaks.

Results This work seems to be about 5-50x faster (depending on whether they train over LAN or WAN) then Secure ML, while achieving comparable or better accuracy on a number of well-known data sets.

3.5 Quantitative verification of neural networks and its security applications

Speaker: Theodora Baluta, National University of Singapore

Neural networks have a variety of issues and vulnerabilities. Theodora strives to verify properties of neural networks. As an example, we look at fairness: Here, we might want that a sensitive feature should not impact the outcome of a neural network, e.g., if the classification depends on the gender of the network. Theodora claims that given a fairness predicate, we can almost always find a counter-example against this predicate.

Since individual such examples don't really tell us much, we ask: "For how many individuals does the prediction not change when there is a change in a sensitive feature?". To answer this, we can, for all samples in our test set, test whether a change in the attribute will change the outcome. This approach, however, doesn't guarantee anything about samples outside of the test set.

We thus move to a quantitative notion of verification. We want to be able to say with high confidence, that the ground truth is very close to the reported values. When using this analysis, we can choose the exact parameters.

How do we quantify this efficiently? One way to do this is by mutating features, one at a time, but this leads to an exponential number of tests. To get better results Theodora encodes this into SAT instances and then uses model counting estimations to get an approximate answer, thus leveraging advances in other areas of computer science. This encoding is a subtle process.

One of the interesting aspects here is that if we have an encoding of a particular nature (equicardinality), then properties of neural networks actually compose.

Results Fairness: They trained 4 binary neural networks on the UCI Adult dataset with 66 binarized input attributes. They encode a few fairness properties and then produce estimates within 6 hours; these estimates seem to deviate from model to model, but appear to be stable in a range of 10-20%.They similarly looked at how robust the models are against adversarial samples, for the MNIST dataset and get some estimates, as well as Trojan attacks (targeted misclassification).

Performance: The tool seemed able to provide estimates for many formulas within 24hrs using at most 4GB RAM

3.6 ABS: scanning neural networks for back-doors by artificial brain stimulation

Speaker: Wen-Chuan Lee, Purdue University

AI and model sharing is becoming increasingly popular. Just like traditional software, models can be Trojans though: they can appear normal, but given specific "triggers", e.g., parts of inputs, they can behave differently.

All previous attacks are in the pixel space with fixed patterns. In this work, they look at feature-space triggers, such as specific Instagram filters. This is interesting, since existing defenses use pixel perturbations to see whether the classification changes. Moreover, such defenses have several limitations, both in terms of a reduction in benign accuracy and in terms of how to apply them.

They observe that trojan models tend to have a weird subspace that they can search for by changing activations per dimension in a linear scanning fashion. They check every single neuron in this way. After identifying potentially compromised neurons, they try to generate a trigger. They then check whether this trigger, when added to different benign inputs, leads to the same (wrong) classification.

Results They look at a range of datasets and 177 models trojaned with a variant of techniques from existing work. It seems that they could reverse-engineer the trojan triggers (somewhat). They score models and seem to be able to distinguish between trojaned models and benign models.

As an amusing anecdote, some benign models had high scores, but there was a good explanation: for these datasets, some of the images were to be categorized as "deer" and the benign models mainly looked for the existence of antlers. If antlers are stamped on any other image, this image might be classified as "deer" too, for these benign models.

Caveat In the questions the speaker admitted that this work assumes that only a single neuron is compromised.

3.7 Lifelong anomaly detection through unlearning

Speaker: Shiqi Shen, National University of Singapore

When trying to identify anomalies, we typically have few training samples that are anomalies among a huge number of samples that are normal. To detect future / unknown anomalies it's better to not require positive data samples to exist.

Given training data with zero or few anomalies, a generative model can learn to generate normal data. We can then compare this artificially generated "normal" data with our actual data to find anomalies.

How do we update the model? If we encounter a false positive, we are aware of that and can update the model. If we encounter a false negative,

things are more complicated: we have to *unlearn* the specific instance from the generator.

We can do that with long-short term memory (LSTM) models. These models look at a sliding window of the last few time steps in the history and tries to predict the next step. We then compare the actual next step with the prediction (using all logits) and consider it abnormal if it is extremely unlikely in our prediction. If that was a false positive, we can update the model to increase the prediction chance of what we just observed.

If, however, the model already predicts a specific outcome, but it turns out that this is an anomaly, we want the model to adjust its prediction for this output to zero. This work proposes to use gradient ascend (the opposite of descent) here, where we force the model to move towards what seems to be a less optimal model (we invert the sign). As we "anti-optimize", we need to put a maximum loss up to which we will learn.

3.8 Seemless end-to-end encrypted messaging with less trust

Speaker: Harjasleen Malvai, Cornell University

When performing end-to-end encryption we might not have a PKI in place. To avoid man-in-the-middle attacks, also called mis-binding, we look at (privacy-preserving) verifiable key directories.

Harjasleen strives for auditable soundness (if the server misbehaves, it can get caught) and privacy in the sense of zero-knowledge.

Methodology

- The server regularly publishes a digest that is small and doesn't leak information about the actual database.
- Now Bob can register with his key, the server checks whether a user with name Bob already exists and if not his key will be added to the next digest.
- When Bob wants to retrieve Alice's key, the server checks whether Bob is blocked by Alice and, if not, sends the key to Bob.
- Bob can check whether Alice's key is correct.
- Users can also update their keys and the updated key will be added to the next digest.
- Bob can call a self-audit function to verify whether his key is correctly stored.
- There's some digest auditors (who could be users) that can call a separate DigestAudit function to make sure the digests are correct somehow.

Technically, they use zero-knowledge sets, which are based on hash-maps. To allow updating of keys without leaking whose key was updated they introduce append-only zero-knowledge sets, basically by adding "Alice — version number" for an increasing version number. The digest auditors not only need to check that nothing is removed from the set. This leaks Alice's version number to her contacts. To avoid users receiving old versions of keys, they add a second zk-set of stale keys that include all keys that have been removed (updated); this is also an append-only zk-set.

This is still slightly problematic as the server could add keys with increased version number (up to the current server epoch). To avoid this, in comes SEAM-less: You have the "all" set and the stale set as before. They add marks (every power of two) that users can check. Alice now only has to check whether a key up to the next checkmark is published by the server, as well as whether any future checkmark up to the current server epoch has been published. These checkmarks are at powers of two, since users aren't expected to change their keys very often, but server epochs are expected to progress quickly. Thus, only a log number of marks need to be checked and it only becomes inefficient for users who change their keys really often.

3.9 PrivDPI: Privacy preserving encrypted traffic inspection with reusable obfuscated rules

Speaker: Jianting Ning, Fujian Normal University, National University of Singapore

This talk is about allowing "middle-boxes" (entities between clients and servers) to be able to perform deep packet inspections over encrypted traffic. We don't want this entity to be able to decrypt and re-encrypt the data, so it needs to work on encrypted data, which is fairly inefficient.

They assume this middle-box to be semi-honest. Either client or server can be malicious.

The payload contains encrypted tokens and the middle-box uses encrypted rules that appear to be pattern-matching between encrypted tokens. These tokens are generated by the client and validated by the server. The middle-box doesn't actually analyze the payload, but only checks the tokens. These tokens can be kept similar. Since either the client or the server is honest, the correctness of these tokens kind of holds by definition.

The questions and answers highlight the weakness of this approach: it assumes that client and/or server assist the middle-box in enforcing any policy. However, if the client was honest, it could simply adhere to the policy and if the server was honest it would not accept violations or collaborate with the middle-box entity. [16]

[16] Comment: The exact use-case envisioned here is not completely clear (to me).

3.10 Updatable anonymous credential systems and applications to incentive systems

Speaker: Jan Bobolz, Paderborn University

Updatable anonymous credentials If we want to update anonymous credentials, we have to show all of our attributes to the credential issuers and we then receive a new credential. This is privacy invasive, as we tell the issuer more than necessary. This work looks at updatable credentials in the following sense:

- User and issuer need to agree on an update function ψ.
- The inputs of ψ are the old attributes A, the output are the new attributes $\psi(A)$ or an error symbol $\perp$.
- After the update, the issuer should have no knowledge about the actual attributes of the user.

Note that the update function can contain conditions; the issuer learns the update function (naturally), but not the attributes (before or after). This way, updates such as increasing the lifetime of a subscription to a service without revealing the current state of the subscription are easily possible.

Incentive systems Users may have loyalty cards or other loyalty point systems that encourage being a loyal customer. If the user ID is revealed during transactions, the user can potentially be tracked and their privacy violated.

To implement this we can use the updatable anonymous credentials from above. Users have a credential with, initially, 0 loyalty points. As the issuer can notice when the update function failed, this works quite directly, except that users now can double-spend points. To avoid this, the user is forced to reveal a double-spending ID that is only used to prevent double-spending. Each credential contains one such ID and the update function is allowed to have a secret user parameter that sets a new (random) double-spending ID. While this solution works if the server is constantly online, it doesn't work otherwise.

Jan now introduces offline double-spending protection: we add a new user secret and a double-spending random value to the credentials. These additional values are used to implement a double-spending prevention that leaks the user's secret key if they try to double-spend. The knowledge of the user secret can now allow the issuer to prove that and which user double-spent.

4 CCS Main Conference, Thursday

4.1 ML Security III – Seeing isn't believing: towards more robust adversarial attack against real world object detectors

Speaker: Yue Zhao, Chinese Academy of Sciences

Autonomous cars rely on correct object detection and errors can be fatal. Yue asks the rhetorical question as to whether adversarial attacks can fool object detection models as well. This work apparently aims at doing just that: robustly attacking object detection models. This is not straight-forward, since differences in angles, distance, position, and illumination make straight-forward attacks difficult.

This work presents an attack that works under the following conditions: long distance ($> 25m$), wide angle ($-60°$ to $60°$), multi-illumination under different environments.

To make their attack more robust, they place their modified stop-sign images into realistic background images found online that already contained a stop sign. This helps them to learn how to mask the sign even in the presence of contextual clues around the actual sign. When considering long distances, the adversarial mask only makes up a few pixels when compared to its effect in short range.

For their evaluation they placed a smartphone into actual driving cars at different speeds to measure whether the attacks would be possible in real-world scenarios. They looked at three types of defenses: modified inputs (e.g., JPEG compression, randomization, etc.), "just training better models", and models trained with adversarial attacks in mind. [17]

4.2 ML Security III – AdVersarial: Perceptual ad blocking meets adversarial machine learning

Speaker: Florian Tramèr, Stanford University

Advertisements are (and in many parts of the world legally are required to) be distinguishable by humans from regular content. In order to make automatic ad-blocking more difficult, content providers and ad-blockers are engaged in an arms race of HTML obfuscation and improved attacks. Even though there is an industry standard for the AdChoices logo, a lot of slight variations of this logo are founds in the wild.

A novel approach to identify ads is to ignore all traditional ways to identify ads and replace them with visual approaches, where, e.g., a neural network is run on a screenshot of the website to identify ads this way. This technique is known as perceptual ad-blocking. If we can create an ad-blocker that can detect ads

[17] Comment: I think that this work pushes the limits of ethical research. While it is important to be aware of potential attacks on object detection in the autonomous car setting, I'm not sure whether showing a road map for breaking these systems under realistic scenarios is worth the insights for the community.

the same way humans can, then ads cannot remain undetected anymore without violating legislation.

In comes this work, to the rescue of the advertisement industry: Florian uses adversarial machine learning to mislead perceptual ad-blockers. These techniques can even be used quite maliciously to block legitimate content (e.g., content from other users).

Threat model In this particular adversarial ML setting, Florian assures, there definitely is an attacker highly motivated to attack the system. Moreover, there are no simpler attacks: here, the attacker wants the human to see the ads (as undisturbed as possible), while still affecting the classifier. We also assume here, that the adversary has access to the ad-blocker, since the ad-blocker is ran client-side.

Florian claims that unless the answer to all these questions (Is there a motivated adversary? Is there no easier way? Is it realistic that there is white-box or black-box access to the model?) is positive, then we shouldn't worry about adversarial machine learning attacks (as the most relevant attack vector).

Here, the attacker has white-box access to the ad-blocker and can prepare attacks offline.

Goals and results False negatives and false positives can be exploited.

- The first goal here is to make ads unrecognizable by the ad-blocker. One possibility for evasion is to have the publisher perturb all rendered pixels of the website. This work showed that at least currently existing ad-blockers could be easily fooled with this approach.
- Another possible goal is to add a honeypot to the website, detect whether it is being blocked and, if so, show a banner to the user, telling them to please deactivate ad-blockers (or refusing to show content).
- Finally, a new goal is privilege abuse: The ad-blocker can be abused by one user's malicious content to have an effect on other parts of the page, e.g., by blocking other user content. The way perceptual ad-blockers are trained is necessarily in a way that allows non-localized effects: Florian explains that ad-blockers need to be trained in a way that allows them to be attacked in such a way, since localized effects lead to other vulnerabilities.

While ad-blockers that emulate human detection of ads would spell doom for the ad industry, Florian is sure that at least with current technology and in the adversarial setting, this is not possible.

4.3 ML Security III – Attacking graph-based classification via manipulating the graph structure

Speaker: Binghui Wang, Duke University

Graph-based classification methods are used in a couple of important settings. This work is the first study on attacking so called collective classification methods. Here, the input(s) to the classifier are a graph of nodes and edges, where some nodes are labeled and the classifier tries to label the remaining ones.

Threat model and goal They look at the attacker's background, such as whether or not the attacker knows the complete graph, the parameters of the training or the training set; as well as on the attackers capabilities, such as whether or not they can add fake edges, remove existing edges

They consider the attack as an optimization problem, where the attacker tries to minimize the total cost of modifying a given graph structure (in terms of edges), while maximizing the number of misclassified target nodes.

As edges are binary, which hurts the optimization, they first convert the binary variables (edge or no edge) into a continuous variable, then solve their optimization, and finally convert the continuous edges back into binary ones. To keep their computations of scores feasible, they apply a few tricks before using projected gradient descent.

Results They looked at existing graphs (subgraphs of Facebook, Twitter, Enron) with synthetic "adversarial nodes". The attacker's target nodes (the ones supposed to be misclassified) are either randomly chosen or a connected component of "adversarial nodes", or nodes that are close to regular nodes.

Different costs can be assigned for edges depending on the respective nodes (either all nodes cost the same, or costs are randomly assigned, or costs depend on the types of nodes). Their baseline approaches consist of either a completely random edge addition and removal, or a simplistic heuristic based on connectivity to adversarial or normal nodes.

While the heuristic was already somewhat effective, their attack outperforms it significantly and reaches $> 95\%$ success in most cases. Moreover, their attack transfers well to other graph neural networks. The attack is very efficient, requiring only a few seconds to run.

4.4 ML Security III – Latent backdoor attacks on deep neural networks

Speaker: Huiying Li, University of Chicago

Huiying starts off strongly, claiming that current backdoors don't apply to modern machine learning due to the usage of transfer learning, before claiming that she will introduce a stronger, resilient backdoor.

A backdoor attack here is a hidden malicious behavior trained into a deep neural network (DNN). The network will behave normally on clean inputs, while deviating significantly when facing specifically modified inputs. These novel

attacks are dangerous, since training DNN's is expensive. Thus companies use transfer learning. For example, starting from Google's highly successful face-recognition model as a teacher model, we can train a student model that we then adapt to our use-case. The main insight of transfer learning is that high-quality features can be re-used, e.g., by removing the last (few) layer(s) of a model and keeping the first layers that extract features from the raw data very successfully.

Since common backdoor attacks are trained back-to-back, the deletion of the last layers breaks the backdoor. Even if the attacker tries to embed a backdoor into a student model, the attacker has a fairly small window-of-opportunity, since training a student model is very efficient and doesn't take a lot of time.

The latent backdoor We assume that the attacker has a potential target class and access to the teacher model.

The latent backdoor now works as follows: we replace the classification layer(s) from the teacher model with a classification layer training on the target as well. We train the model to inject the backdoor. Finally, we throw away the classification layer and put the old one back in.

This type of backdoor, Huiying assures us, survives transfer learning, is much harder to detect and can affect all student models derived from the target model. Moreover, we could prepare future attacks with, say, several potential targets.

Evaluation The effectiveness of this approach varies on further assumptions: If the attacker can predict what the student data of the target will be (all the images), they achieve a very high success rate (96%+) without harming the model's accuracy. If the attacker correctly guesses or knows at least one picture of the target (that will be used), the attack is still fairly good. Finally, they looked at the scenario where the attacker uses distinct images of the target for the attack.

Defenses Neural Cleanse (S&P 2019) and Fine-Pruning (RAID 2018) did not work against this attack. Input image blurring also didn't seem effective: in order to disrupt the trigger, the normal accuracy had to be degraded significantly.

An effective defense against their specific attack seemed to be multi-layer tuning: Instead of tuning the final classification layer only for the transfer learning, they tuned some of the already trained (copied) layers as well. [18]

[18] Comment: Their specific attack was disrupted by just re-training one of the copied layers, I think that this could be easily defeated by applying the same latent backdoor technique with shifting the last layer(s) by one: one more layer is thrown away during training of the teacher model. This came up in the Q&A session as well and is partially covered in the paper. Apparently their attack's success dropped very significantly when trying to move to earlier layers and the exact architecture of the model seems to be important as well.