On Primes, Log-Loss Scores and (No) Privacy

Abhinav Aggarwal
Amazon Alexa
Seattle, WA USA
aggabhin@amazon.com

Zekun Xu
Amazon Alexa
Seattle, WA USA
zeku@amazon.com

Oluwaseyi Feyisetan
Amazon Alexa
Seattle, WA USA
sey@amazon.com

Nathanael Teissier
Amazon Alexa
Arlington, VA USA
natteis@amazon.com

Abstract

A common metric for assessing the performance of binary classifiers is the Log-Loss score, which is a real number indicating the cross entropy distance between the predicted distribution over the labels and the true distribution (a point distribution defined by the ground truth labels). In this paper, we show that a malicious modeler, upon obtaining access to the Log-Loss scores on its predictions, can exploit this information to infer all the ground truth labels of arbitrary test datasets with full accuracy. We provide an efficient algorithm to perform this inference.

A particularly interesting application where this attack can be exploited is to breach privacy in the setting of Membership Inference Attacks. These attacks exploit the vulnerabilities of exposing models trained on customer data to queries made by an adversary. Privacy auditing tools for measuring leakage from sensitive datasets assess the total privacy leakage based on the adversary's predictions for datapoint membership. An instance of the proposed attack can hence, cause complete membership privacy breach, obviating any attack model training or access to side knowledge with the adversary. Moreover, our algorithm is agnostic to the model under attack and hence, enables perfect membership inference even for models that do not memorize or overfit. In particular, our observations provide insight into the extent of information leakage from statistical aggregates and how they can be exploited.

1 Introduction

Protecting customer privacy is of fundamental importance when training ML models on sensitive customer data. While explicit data de-identification and anonymization mechanisms can help protect privacy leakage to some extent, research has shown that this leakage can happen when models trained on customer data can be queried by an external entity (Homer et al., 2008; Sankararaman et al., 2009; Li et al., 2013; Shokri et al., 2017), or when statistical aggregates on the dataset are exposed (Dwork and Naor, 2010; Dwork et al., 2017).

Recently, it was shown that the knowledge of Log-Loss scores leaks information about true labels of test datapoints under some constraints on the prior knowledge on these labels (Whitehill, 2018). However, extracting meaningful information from these aggregates on arbitrary large datasets, while maintaining reasonable inference accuracy in a limited number of queries to a Log-Loss oracle remained an open problem, specially in cases when no prior knowledge is available. Moreover, the number of queries required by their algorithm scales with the size of the test dataset. We address this problem in this paper and provide multiple algorithms for optimal inference of arbitrarily many test labels in a single query using the exposed Log-Loss scores. This sheds insight into the extent of information leakage from this statistical aggregate and how it can be exploited to game a classification task, for example, in the context of data-mining competitions like Kaggle, KDDCup and ILSVRC Challenge (Russakovsky et al., 2015).

More concretely, consider the following scenario: you are tasked with a critical binary classification problem. The quality of your solution will be assessed through a performance score (Log-Loss) on an unknown test dataset. If you score the highest among all candidate solutions, then you win a significant cash prize. You are allowed only two attempts at the solution and the best of the two scores will be considered.

Is it possible to game this system in a way that your score is always the highest amongst all candidates, without even training any classifier?

We answer this in the affirmative by showing that the knowledge of only the size of the test dataset

is enough to construct a scheme that can game any binary classifier that uses the Log-Loss metric to assess the quality of classification. This scheme is completely agnostic of the underlying classification task and hence, sheds light on how a malicious modeller can fake a perfect classifier by demonstrating zero test error. We assume that the oracle reports the scores truthfully on the entire dataset.

A particularly interesting application of our observation is for breaching membership privacy, where an attacker can query the model for inference on a set of datapoints and use these responses to infer what datapoints were used to train that model. Given blackbox access to a model and a data point x, this attack model is a binary classifier to infer the membership of x in the training dataset of the target model using its output on x – the more information this output reveals, the better this inference can be performed. Consequently, the accuracy of the attack depends on how well the adversary can capture the difference in model performance. Nonetheless, the popularity of this attack has made it a strong candidate for assessing privacy leakage of models trained on datasets containing sensitive information (Song and Shmatikov, 2019; Backes et al., 2016; Pyrgelis et al., 2017; Salem et al., 2018; Liu et al., 2019; Murakonda and Shokri, 2020). A successful attack can compromise the privacy of the users that contribute to the training dataset. Our results show that an oracle access to Log-Loss scores (for example, when using privacy auditors on sensitive datasets) enables full privacy breach in a single query[1].

Related Work In recent work, (Blum and Hardt, 2015) demonstrated how an attacker can estimate the test set labels in a competition setting with probability $2/3$. Similarly, and more related to our work, (Whitehill, 2016, 2018) showed how the knowledge of AUC and Log-Loss scores can be used to make inference on similar test sets by issuing multiple queries for these statistics. Our work extends the latter to optimize the number of queries. Similarly, through a Monte Carlo algorithm, (Matthews and Harel, 2013) show how knowing most of the test labels can help estimate the remaining labels upon gaining access to an empirical ROC curve. However, their algorithm is far from exact inference with no *apriori* information of the true labels.

We further observe that the theme of our work is related to two fields of research: adaptive data analysis, and protections of statistical aggregates using Differential Privacy (DP). In adaptive data analysis (Hardt and Ullman, 2014; Dwork et al., 2015), an attacker leverages multiple (adaptive) queries to sequentially construct a complete exploit (e.g., of a test set). Conversely, with DP (Dwork et al., 2006) the objective is to protect the aggregate statistics, such as those exploited by (Whitehill, 2016), from leaking information.

Log-Loss Metric We begin with reminding the reader of the definition of the Log-Loss metric on a given prediction vector with respect to a binary labeling of the datapoints in the test dataset.

Definition 1 (Log-Loss)**.** *For a dataset* $D = [d_1, \ldots, d_{|D|}]$*, let* $\ell \in \{0,1\}^{|D|}$ *be a binary labeling and* $\mathbf{x} = [x_1, \ldots, x_{|D|}] \in [0,1]^{|D|}$ *be a vector of prediction scores. Let* $g(\ell_i, x_i) = \ell_i \log_e x_i + (1-\ell_i)\log_e(1-x_i)$*. Then, the Log-Loss (LL in short) for* $\mathbf{x}$ *with respect to* ℓ *is defined as* $LL(\mathbf{x}, \ell) = -\frac{1}{|D|}\sum_{i=1}^{|D|} g(\ell_i, x_i)$.

The definition easily generalizes for multi-class classifiers. A common variant is to ignore the normalization by $|D|$. Our constructions in this paper are scale-invariant.

2 Algorithms for Exact Inference using Log-Loss scores

In this section, we discuss multiple algorithms for single shot inference of all ground-truth labels using carefully constructed prediction vectors that help establish a 1-1 correspondence of the Log-Loss scores with the labelings of the test dataset. We, therefore, refer to the entity that performs such an inference as an *adversary*.

2.1 Inference using Twin Primes

Our first algorithm uses twin-primes, i.e. pairs of prime numbers within distance 2 of each other (see OEIS A001359 from https://oeis.org/A001359.). It has been conjectured that infinitely many such pairs exist (de Polignac, 1851). For a dataset of some finite size $|D| \geq 1$, we require $|D|$ such pairs. The main steps of our approach are

[1] Our paper can be viewed from two different lenses: exploiting, and information with the intent of protection. On one hand, our results demonstrate how to exploit performance metrics that are routinely vended from ML models, while on the other hand, they call to attention these vulnerabilities which might be currently under silent exploitation. Armed with this information, individuals and organizations which vend these seemingly innocuous aggregate metrics from their classification models, can grasp the potential scope of the information leakage that can result from this.

Algorithm 1: Inference on dataset $D = [d_1, \ldots, d_{|D|}]$ using Twin Primes

1 Let $5 \leq p_1 < \cdots < p_{|D|}$ be a sequence of (smallest) primes such that $p_i + 2$ is also a prime for all i.
2 Form the prediction vector for D as $\mathbf{x} = \left[\frac{p_1}{2+p_1}, \ldots, \frac{p_{|D|}}{2+p_{|D|}}\right]$.
3 Obtain the Log-Loss on $\mathbf{x}$ and use that to infer the ground-truth labels for D using Algorithm 2.

Algorithm 2: True Labels from Log-Loss

1 **Input:** Log-Loss score s
2 **Output:** True Labels for datapoints in D
3 Let $e^{s|D|} = p/q$ (lowest form) and $q = 2^m p_1 \cdots p_k$.
4 Find the set of (zero-indexed) locations I of primes $p_1 \ldots p_k$ in OEIS A001359.
5 Construct the labeling ℓ as follows: Insert 1 in indices specified by I, and 0s elsewhere.
6 Return ℓ.

outlined in Algorithm 1 and the following theorem proves its correctness.

Theorem 1. *If the Twin-Prime Conjecture holds, then for any dataset D, the Log-Loss scores returned by Algorithm 1 are in 1-1 correspondence with the binary labelings for datapoints in D.*

Proof. For a fixed D and labeling ℓ, it suffices to show that $-|D| \cdot LL(\mathbf{x}, \ell)$ takes all unique values. From Definition 1, it holds that:

$$-|D| \cdot LL(\mathbf{x}, \ell) = \log_e \left(\frac{2^{|D_0|} \prod_{d_j \in D_1} p_j}{(2+p_1) \cdots (2+p_{|D|})} \right).$$

Now, fix any two labelings ℓ_1 and ℓ_2 for D. If the number of zeros in them are different, then it is easy to see that $\mathbf{x}$ will give different Log-Loss scores on both of them, since the exponent of 2 in the numerator will be different for these two labelings and all other prime numbers being odd in the denominator, no common factors will exist to cancel this effect. If the number of zeros is the same, then observe that the following holds:

$$|D| \left(LL(\mathbf{x}, \ell_1) - LL(\mathbf{x}, \ell_2) \right) = \log_e \frac{\prod_{d_i \in D_1^{(2)}} p_i}{\prod_{d_j \in D_1^{(1)}} p_j},$$

where $D_1^{(2)}$ is the set of datapoints with label 1 in ℓ_2 (similarly for $D_1^{(1)}$). Now, since ℓ_1 and ℓ_2 are different, there must exist some index $1 \leq k \leq |D|$ for which $\ell_1(k) = 0$ and $\ell_2(k) = 1$. Thus, p_k will appear in the numerator but not in the denominator. Moreover, since the denominator is also a product of primes, it does not divide the numerator in this case, and hence, the difference $LL(\mathbf{x}, \ell_1) - LL(\mathbf{x}, \ell_2)$ is non-zero. □

As an example of this technique, assume $|D| = 2$. The primes we can use for this construction are 5 and 11, so that the prediction vectors can be set as $v_1 = [5/7, 2/7]$ and $v_2 = [11/13, 2/13]$. Then, the following lists the log-loss values for the prediction vector $\mathbf{x}^* = [5/7, 11/13]$ (as per Algorithm 1):

$$-2LL(\mathbf{x}^*, [0,0]) = \log_e \frac{2}{7} + \log_e \frac{2}{13} = \log_e \frac{4}{91}$$
$$-2LL(\mathbf{x}^*, [0,1]) = \log_e \frac{2}{7} + \log_e \frac{11}{13} = \log_e \frac{22}{91}$$
$$-2LL(\mathbf{x}^*, [1,0]) = \log_e \frac{5}{7} + \log_e \frac{2}{13} = \log_e \frac{10}{91}$$
$$-2LL(\mathbf{x}^*, [1,1]) = \log_e \frac{5}{7} + \log_e \frac{11}{13} = \log_e \frac{55}{91}$$

Our construction allows us to give an algorithm to determine the true labeling from the Log-Loss value, without having to consult a lookup table (see Algorithm 2). This follows from Gauss's Fundamental Theorem of Arithmetic (GFoA), that every positive integer is either a prime or is uniquely factorizable as a product of primes (Gauss, 1966). We assume that the Log-Loss score s is reported such that $e^{s|D|} = p/q$ is a rational number in its reduced form (i.e. with $q \neq 0$ and $\gcd(p, q) = 1$), and that, without loss of generality, the prediction vector was constructed using the first $|D|$ prime numbers, as specified in Algorithm 1.

As an example, suppose on a dataset of size 3, the Log-Loss s is reported such that $e^{3s} = 1729/170$. Note that this requirement of knowing $|D|$ is not necessary, since it is equal to the number of factors of the numerator of $e^{s|D|}$. Now, writing the denominator $170 = 2^1 \times 5 \times 17$, we note that there is 1 zero in the labeling, and the other two labels are one. From OEIS A001359, we note that 5 and 17 are the first and third prime numbers in the series (when we start counting from 5), respectively, and hence, the first and third datapoints must have labels one. Thus, we have inferred that the true labeling for D must be $[1, 0, 1]$.

2.2 Extension to Multiple Classes

A similar construction can be used to infer all true labels in a multi-class setting as well. For the One-vs-All approach, then it is trivial to see that the individual Log-Loss scores for each class reveal datapoints from that class. For the K-ary classifier approach (K being the number of classes), the following construction works: Let $p_1, \ldots, p_{|D|}$ be the first $|D|$ primes. For datapoint d_i, use the following prediction vector: $v_i = \left[1/\alpha_i, p_i/\alpha_i, \ldots, p_i^{K-1}/\alpha_i\right]$, where $\alpha_i = \sum_{j=0}^{K-1} p_i^j$, thus, forming the prediction matrix $v_D = [v_1, \ldots, v_{|D|}]$. Given the true labels $\ell \in \{1, \ldots, K\}^{|D|}$, it can be shown that the following holds:

$$-LL(v_D, \ell) + \sum_{j=1}^{K} \log_e \alpha_j = \log_e p_1^{\ell_1 - 1} \cdots p_{|D|}^{\ell_{|D|}-1}.$$

This gives the required injection, since the sum on the left is constant for fixed K and $|D|$, and the product on the right is unique (following GFoA).

2.3 Inference using Binary Representations

In Algorithm 1, the main reason why we chose distinct primes was that when the denominator of $e^{s|D|}$ was factorized, the prime factors would uniquely define the locations of 1s in the binary labeling. The same 1-1 correspondence can be achieved by observing that the each binary labeling is also equivalent to a binary representation (base 2) of a natural number (see Algorithm 3). By using powers of 2 for only the indices corresponding to locations of 1s in the binary labeling, when the denominator is now factorized, it produces in the exponent of the 2 an integer, whose binary representation (when reversed) is exactly the same as the labeling. This also helps eliminate the dependence on the Twin Prime Conjecture. The following theorem establishes the correctness of our algorithm.

Theorem 2. *For any dataset D, the Log-Loss scores returned by Algorithm 3 are in 1-1 correspondence with the labelings for datapoints in D.*

Proof. Similar to the proof of Theorem 1, for a fixed D and labeling ℓ, it suffices to show that $d = |D| \cdot LL(\mathbf{x}, \ell)$ takes all unique values. Let I_1 be the set of indices in ℓ that have value 1. Now, since $x_i = \frac{2^{2^{i-1}}}{1+2^{2^{i-1}}}$, we can write the following:

$$d = \sum_{j=1}^{|D|} \log_e \left(1 + 2^{2^{j-1}}\right) - \frac{\sum_{i \in I_1} 2^{i-1}}{\log_2 e}.$$

Algorithm 3: Exact Inference using Binary Representations for $D = [d_1, \ldots, d_{|D|}]$.

1 Let $\alpha_i = 2^{2^{i-1}}$. Form the prediction vector for D as $\mathbf{x} = \left[\frac{\alpha_1}{1+\alpha_1}, \ldots, \frac{\alpha_{|D|}}{1+\alpha_{|D|}}\right]$.

2 Obtain the Log-Loss on $\mathbf{x}$ and use that to infer the true labels for D.

Thus, if $LL(\mathbf{x}, \ell_1) = LL(\mathbf{x}, \ell_2)$ for two distinct labelings ℓ_1 and ℓ_2, then from above, it is easy to see that this can only happen when $\sum_{i \in I_1^{(1)}} 2^{i-1} = \sum_{i \in I_1^{(2)}} 2^{i-1}$, where $I_1^{(j)}$ is the index set (similar to I_1) for labeling ℓ_j. Now, since every positive integer has a unique binary representation, this implies that $I_1^{(1)} = I_1^{(2)}$, which can only happen when the two labelings are the same. Moreover, note that since powers of 2 are always even, the product in the denominator of the equation above has no common factors with the numerator. Thus, each binary labeling of $\mathbf{x}$ gives a unique Log-Loss score. □

As an example, if the true labels for a dataset D containing four datapoints are $[1, 0, 1, 1]$, the exponent of 2 can be the natural number represented using the binary representation 1101, which is 13. Similarly, if the exponent observed is, say 18, then the corresponding binary representation is 10010, and hence, the true labels must be $[0, 1, 0, 0, 1]$.

3 Adapting to Fixed Precision Arithmetic

Can we design prediction vectors such that the Log-Loss scores are atleast some Δ apart from each other, where Δ is limited by the floating point precision on the machine used to simulate our inference algorithms?

For distinguishing scores with ϕ significant digits, since there are a total of 10^ϕ possible numeric values, the threshold value of separation is $\Delta \geq 10^{-\phi}$. If the separation in the scores is smaller than this value, then they cannot be distinguished. Inverting this inequality gives $\phi \geq \left\lceil \log_{10}\left(\frac{1}{\Delta}\right) \right\rceil$. For example, if one wishes to have the scores separated by $\Delta \geq 0.2$, then the minimum amount of precision required is $\lceil \log_{10} 5 \rceil = 1$. For $\Delta = 0.002$, we would need $\phi \geq \lceil \log_{10} 500 \rceil = 3$ digits.

We can reduce the requirement of a large precision by combining the AUC and Log-Loss scores, which is common in most practical situations where multiple performance metrics are evaluated to give a holistic overview of classifier inference. This

way, even if they are individually not-unique but the tuple is unique for each labeling, exact inference can be done. For example, consider a dataset $D = [d_1, d_2, d_3]$ and the prediction vector $v = [0.2, 0.4, 0.6]$. Clearly, neither the AUC scores nor the Log-Loss scores are unique. However, if we consider the two scores together, the labels can be uniquely identified. Moreover, precision of only two significant digits is enough to make this decision.

A rough analysis tells us that with ϕ significant digits, there are $10^{\phi}(10^{\phi}+1)$ possible unique values in the AUC-Log-Loss tuple (the +1 is to take into account the case when AUC is Not-Defined). Using the pigeonhole principle, for any dataset D with $|D| = n$, a necessary condition for unique inference is that $2^n \leq 10^{\phi}(10^{\phi}+1)$, which gives $\phi \gtrsim \lceil 0.151n \rceil$. Conversely, with a precision of ϕ significant digits, one can only hope to uniquely identify labels for datasets of size at most $\left\lfloor \log_2 \left(10^{\phi}(10^{\phi}+1)\right) \right\rfloor \leq 7\phi$. We can recurse over the remaining points in the database in this situation, for exact inference in at most $\lceil n/6\phi \rceil$. For example, using the IEEE 754 double-precision binary floating-point format, which has at least 15 digits precision, at most $\left\lceil \frac{|D|}{90} \right\rceil$ queries suffice. In particular, 2 queries suffice for all datasets containing fewer than 180 datapoints.

4 Exact Membership Inference Attacks using a Log-Loss Oracle

Our observations provide insight into the extent of information leakage from statistical aggregates and how they can be exploited. A particularly interesting application is for designing stronger Membership Inference Attacks.

These attacks were first proposed to exploit the vulnerabilities of exposing models trained on customer data to queries by an adversary (Shokri et al., 2017), and later proposed to assess (1) data provenance by training language models on sensitive datasets (Song and Shmatikov, 2019) and (2) leakage from model outputs (Murakonda and Shokri, 2020). Such auditing techniques are becoming increasingly popular to quantify privacy leakage and because the underlying attack model is a binary classifier, aggregates like the Log-Loss scores will indicate the total privacy leakage based on the adversary's predictions.

The threat model we are interested in is the case when these aggregates are exposed to a malicious entity (adversary), in which case our analysis demonstrates that this additional information enables the adversary to craft malicious prediction vectors. The response to this query helps infer exactly which datapoints were used for training. Note that the adversary never queries the model under attack directly for prediction on any datapoints whatsoever. Moreover, the interaction with the model curator is similar to the interaction with the model interface in the attack proposed by (Shokri et al., 2017) in that the adversary seeks answers to queries that can help leak information about the training data. The only difference is the additional access to a Log-Loss oracle, which helps make our attack purely deterministic.

5 Conclusion and Future Work

In this paper, we demonstrated how a single Log-Loss query can enable exact inference of ground-truth labels of any number of test datapoints. This highlights the extent of information leakage from statistical aggregates, and in particular, that improperly constructed auditing tools may leak more information about the dataset than the leaky model they seek to expose. We see our work as a caution for such tools against reporting sensitive metrics.

An interesting question to ask is if other popular metrics (like precision, recall, AUC) used in the ML literature can be exploited for privacy leakage in a similar manner. In (Whitehill, 2019), an AUC-ROC oracle on the test dataset is used to deduce the true labels in at most $2|D|$ queries. This opens up opportunities to explore if exact inference on all datapoints is possible with one AUC query.

Yet another interesting question to ask is if exact inference is possible when the adversary learns only a bound on or an approximate value in each Log-Loss query it issues. Observe that by deciding the size of the test dataset, the adversary also fixes the number of possible values Log-Loss scores can take. In a typical scenario where the adversary has some prior knowledge about the amount by which the reported score differs from the actual value (see (Dwork et al., 2019) for an approach to add noise to the reported scores), this discrete set of possible scores can present a huge advantage – the adversary can perform inference over the most likely score under the constraint above. Nonetheless, a distribution over the labelings for the test set can be learnt to bound the inference error.

References

Michael Backes, Pascal Berrang, Mathias Humbert, and Praveen Manoharan. 2016. Membership privacy in microrna-based studies. In *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*, pages 319–330.

Avrim Blum and Moritz Hardt. 2015. The ladder: A reliable leaderboard for machine learning competitions. *arXiv preprint arXiv:1502.04585*.

Cynthia Dwork, Vitaly Feldman, Moritz Hardt, Toniann Pitassi, Omer Reingold, and Aaron Leon Roth. 2015. Preserving statistical validity in adaptive data analysis. In *Proceedings of the forty-seventh annual ACM symposium on Theory of computing*, pages 117–126.

Cynthia Dwork, Nitin Kohli, and Deirdre Mulligan. 2019. Differential privacy in practice: Expose your epsilons! *Journal of Privacy and Confidentiality*, 9(2).

Cynthia Dwork, Frank McSherry, Kobbi Nissim, and Adam Smith. 2006. Calibrating noise to sensitivity in private data analysis. In *Theory of cryptography conference*, pages 265–284. Springer.

Cynthia Dwork and Moni Naor. 2010. On the difficulties of disclosure prevention in statistical databases or the case for differential privacy. *Journal of Privacy and Confidentiality*, 2(1).

Cynthia Dwork, Adam Smith, Thomas Steinke, and Jonathan Ullman. 2017. Exposed! a survey of attacks on private data.

Carl Friedrich Gauss. 1966. *Disquisitiones arithmeticae*, volume 157. Yale University Press.

Moritz Hardt and Jonathan Ullman. 2014. Preventing false discovery in interactive data analysis is hard. In *2014 IEEE 55th Annual Symposium on Foundations of Computer Science*, pages 454–463. IEEE.

Nils Homer, Szabolcs Szelinger, Margot Redman, David Duggan, Waibhav Tembe, Jill Muehling, John V Pearson, Dietrich A Stephan, Stanley F Nelson, and David W Craig. 2008. Resolving individuals contributing trace amounts of dna to highly complex mixtures using high-density snp genotyping microarrays. *PLoS Genet*, 4(8):e1000167.

Ninghui Li, Wahbeh Qardaji, Dong Su, Yi Wu, and Weining Yang. 2013. Membership privacy: a unifying framework for privacy definitions. In *Proceedings of the 2013 ACM SIGSAC conference on Computer & communications security*, pages 889–900.

Gaoyang Liu, Chen Wang, Kai Peng, Haojun Huang, Yutong Li, and Wenqing Cheng. 2019. Socinf: Membership inference attacks on social media health data with machine learning. *IEEE Transactions on Computational Social Systems*, 6(5):907–921.

Gregory J Matthews and Ofer Harel. 2013. An examination of data confidentiality and disclosure issues related to publication of empirical roc curves. *Academic radiology*, 20(7):889–896.

Sasi Kumar Murakonda and Reza Shokri. 2020. Ml privacy meter: Aiding regulatory compliance by quantifying the privacy risks of machine learning. *arXiv preprint arXiv:2007.09339*.

Prince Alphonse de Polignac. 1851. *Recherches nouvelles sur les nombres premiers*.

Apostolos Pyrgelis, Carmela Troncoso, and Emiliano De Cristofaro. 2017. Knock knock, who's there? membership inference on aggregate location data. *arXiv preprint arXiv:1708.06145*.

Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, et al. 2015. Imagenet large scale visual recognition challenge. *International journal of computer vision*, 115(3):211–252.

Ahmed Salem, Yang Zhang, Mathias Humbert, Pascal Berrang, Mario Fritz, and Michael Backes. 2018. Ml-leaks: Model and data independent membership inference attacks and defenses on machine learning models. *arXiv preprint arXiv:1806.01246*.

Sriram Sankararaman, Guillaume Obozinski, Michael I Jordan, and Eran Halperin. 2009. Genomic privacy and limits of individual detection in a pool. *Nature genetics*, 41(9):965–967.

Reza Shokri, Marco Stronati, Congzheng Song, and Vitaly Shmatikov. 2017. Membership inference attacks against machine learning models. In *2017 IEEE Symposium on Security and Privacy (SP)*, pages 3–18. IEEE.

Congzheng Song and Vitaly Shmatikov. 2019. Auditing data provenance in text-generation models. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 196–206.

Jacob Whitehill. 2016. Exploiting an oracle that reports auc scores in machine learning contests. In *Thirtieth AAAI Conference on Artificial Intelligence*.

Jacob Whitehill. 2018. Climbing the kaggle leaderboard by exploiting the log-loss oracle. In *Workshops at the Thirty-Second AAAI Conference on Artificial Intelligence*.

Jacob Whitehill. 2019. How does knowledge of the auc constrain the set of possible ground-truth labelings? In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 5425–5432.