Differential Privacy & Approximate Bayesian Inference

Mrinank Sharma

Supervisor: Dr Richard E. Turner

Department of Engineering
University of Cambridge

This report is submitted for the degree of
Master of Engineering

Pembroke College May 2019

Abstract

ms2314: Need to write this

Table of contents

Chapter 1

Introduction

Machine learning methods are trained on large quantities of data, leveraging information within the dataset in order to make predictions about previously unseen data and make decisions. Recently, such methods have found use in scenarios where the data used in personal and sensitive, one example being the use of human genomic data to predict drug sensitivity [1]. Typically, data relating to individuals in training datasets are *anonymised*, for example, by removing all identifiable information (such as names, addresses, etc) and replacing this information with an anonymous identifier. However, Narayanan and Shmatikov show that anonymisation is insufficient, partially due to the availability of *auxiliary information* i.e. additional, publicly available information. When Netflix released a dataset in 2006 containing movie ratings for approximately 500000 subscribers with names replaced with identifiers, the data could be combined with public ratings on Internet Movie Database to identify movie ratings of two users. [2] Intuitively, data-points about individuals are highly dimensional meaning that anonymisation is insufficient, a further example being that even when sharing DNA sequence data without identifiers, it is possible to recover particular surnames using additional metadata. [3]

ms2314: Look into these papers more

Whilst a particular user's film ratings are not particularly sensitive, a lack of privacy in the areas of healthcare and public policy are critical.

Fredrikson et. al have shown that *model inversion attacks* are possible, where an adversary seeks to learn information about training data given model predictions. In particular, a neural network for facial recognition which returned confidence values was exploited in order to recover the image of a training set participant. Whilst more sophisticated attacks will be required for algorithms which do not provide confidence information, it is therefore possible for an adversary to recover anonymised training data-points and then de-anonymise this information using auxiliary information. [4]

The increasing availability and affordability of mobile smart-phones means that the *federated learning* context, where data across a number of clients is used to train a global model

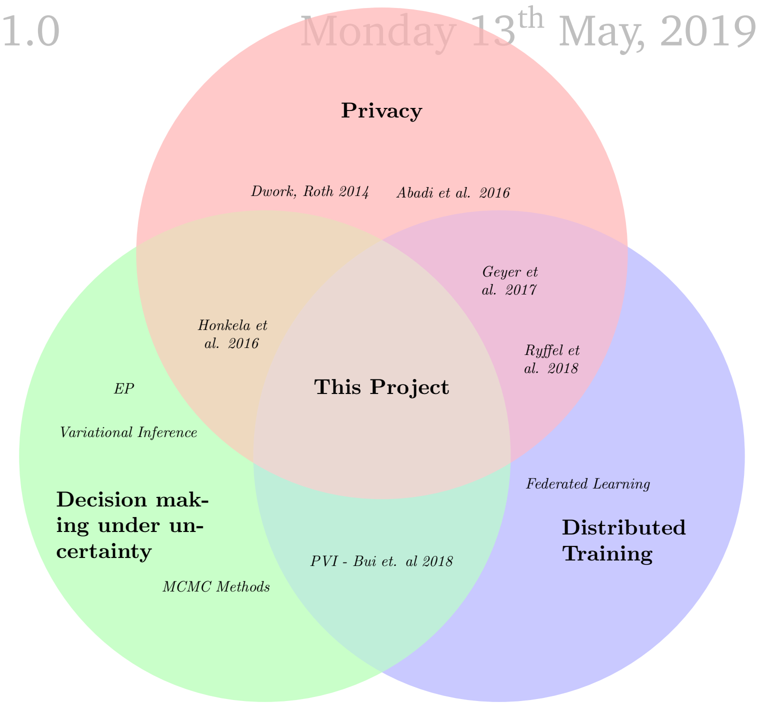

Fig. 1.1 Project aims, including other work within this area.

without transferring local data to a central server, is particularly interesting. Additionally, federated learning schemes reduce power consumption and intuitively give stronger privacy by removing the requirement of transferring entire local datasets. [5] Additionally, in order to make optimal decisions, it is important to model the uncertainty in what is known. This corresponds to performing Bayesian inference and producing a *posterior distribution* over unknown variables.

ms2314: Update Venn Diagram - also add these papers in the references of this report

This project aims to develop a generalised method to enable Bayesian inference to be performed in contexts where data is distributed over a number of clients whilst also providing privacy guarantees for each client.

Chapter 2

Literature Review

2.1 Differential Privacy

2.1.1 Preliminaries

Differential privacy is a mathematical technique which formalises privacy and is able to numerically quantify the level of privacy that some method provides. This is particularly useful not only from the point of view of a designer who is able to compare techniques formally but also from the point of view of a client whose data we seek to protect as this formalism enables them to choose particular settings corresponding to the level of privacy that they seek.

ms2314: Perhaps add diagram showing privacy barrier / we generally assume that the adversary is able to do anything to the data after some sort of model has been released.

Definition 2.1.1 (ϵ-Differential Privacy) *A randomized algorithm, $\mathcal{A}$, is said to be ϵ-differentially private if for any possible subset of outputs, S, and for all pairs of datasets, $(\mathcal{D}, \mathcal{D}')$, which differ in one entry only, the following inequality holds:*

$$Pr(\mathcal{A}(\mathcal{D}) \in S) \leq e^{\epsilon} Pr(\mathcal{A}(\mathcal{D}') \in S) \tag{2.1}$$

Thus, differential privacy provides privacy in the sense that the output probability densities ought to be similar (i.e. bounded by e^{ϵ}) for datasets which are also similar (i.e. differ in only entry only). ϵ, a positive quantity, quantifies the level of privacy provided; large values of epsilon allow the resulting output densities to differ significantly whilst small values of epsilon mean that the output densities are similar. [6]

The key intuition behind differential privacy, noting the requirement that the algorithm is **randomized**, is to introduce noise which obscures the contribution of any particular data-point meaning that any adversary is unable to determine whether a particularly output was simply due to noise or due to a specific data-point.

Often, the above definition of differential privacy is slackened by introducing an extra privacy variable.

Definition 2.1.2 ((ϵ, δ)-Differential Privacy) . *A randomized algorithm, $\mathcal{A}$, is said to be (ϵ, δ) differentially private if for any possible subset of outputs, S, and for all datasets, $(\mathcal{D}, \mathcal{D}')$, which differ in one entry only, the following inequality holds:*

$$Pr(\mathcal{A}(\mathcal{D}) \in S) \leq e^{\epsilon} Pr(\mathcal{A}(\mathcal{D}') \in S) + \delta \tag{2.2}$$

Similar to ϵ, larger values of δ correspond to weaker privacy guarantees and $\delta = 0$ corresponds to a pure ϵ-differentially private algorithm. Note that it can be shown that (ϵ, δ)-DP provides a probabilistic ϵ-DP guarantee with probably $1-\delta$. Additionally, it can be shown that differential privacy is immune to *post-processing* i.e. without additional knowledge, it is not possible to reduce the level of privacy provided by a differentially private algorithm. [6]

A useful quantity relating to a function of some dataset is the ℓ_2 sensitivity.

Definition 2.1.3 (ℓ_2 Sensitivity) *The ℓ_2 sensitivity of function $f : \mathcal{D} \rightarrow \mathbb{R}^n$, is denoted as $\Delta_2(f)$ and is defined as:*

$$\Delta_2(f) = \max_{\mathcal{D}, \mathcal{D}'} ||f(\mathcal{D}) - f(\mathcal{D}')||_2 \tag{2.3}$$

where $\mathcal{D}$ and $\mathcal{D}'$ differ in one entry only.

For a function with a given ℓ_2 sensitivity, the Gaussian mechanism can be used to provide an (ϵ, δ)-differential privacy guarantee.

Theorem 2.1.4 (Gaussian Mechanism) *Let $f : \mathcal{D} \rightarrow \mathbb{R}^n$ be a function with ℓ_2 sensitivity $\Delta_2(f)$. Releasing $f(\mathcal{D}) + \eta$ is (ϵ, δ)-differentially private where $\eta \sim \mathcal{N}(0, \sigma^2)$ when:*

$$\sigma^2 > 2 \ln(1.25/\delta) \Delta_2^2(f)/\epsilon^2 \tag{2.4}$$

[6]

Typically, a particular process may be repeatedly applied to a dataset; a simple example being that in many machine learning problems, a loss function can be composed into a sum over data-points and gradient descent techniques repeatedly compute the gradient of the loss over all of the data-points, repeatedly updating the gradient to reduce the loss. [7] Therefore, it is essential to be able to *compose* the total loss of privacy (in terms of an overall value for ϵ and δ) when a randomised algorithm is repeatedly applied.

Fortunately, there exist many schemes which allow values of ϵ and δ to be computed. The *Moments Accountant* is an advanced technique which is able to 'account' for and track the total privacy expenditure of a technique by tracking the moments of the privacy loss.

2.1.2 The Moments Accountant

Definition 2.1.5 (Privacy Loss) *For neighbouring datasets, $\mathcal{D}$ and $\mathcal{D}'$, a stochastic algorithm, $\mathcal{A}$ and some outcome, S, define the privacy loss at S as*

$$c(S;\mathcal{A},\mathcal{D},\mathcal{D}') \triangleq \log \frac{\Pr[\mathcal{A}(\mathcal{D})=S]}{\Pr[\mathcal{A}(\mathcal{D}')=S]} \tag{2.5}$$

Intuitively, the privacy loss is large when the probability of the observed outcome is very different across neighbouring datasets, corresponding to a large absolute value of c. Note that since the algorithms we use are stochastic, the privacy loss is a random variable. [8]

Definition 2.1.6 (λth Moment of $\mathcal{A}$) *For algorithm $\mathcal{A}$, a quantity of interest is the maximum value of the log of the moment generating function of the privacy loss.*

$$\alpha_{\mathcal{A}}(\lambda) \triangleq \max_{\mathcal{D},\mathcal{D}'} \log \mathbb{E}_{S\sim\mathcal{A}(\mathcal{D})}\left\{\exp \lambda c(S;\mathcal{A},\mathcal{D},\mathcal{D}')\right\} \tag{2.6}$$

[8]

Theorem 2.1.7 (Properties of $\alpha_{\mathcal{A}}(\lambda)$) *$\alpha_{\mathcal{A}}(\lambda)$ exhibits two important properties.*

1. ***Composability:*** *If the algorithm $\mathcal{A}$ consists of a sequence of adaptive mechanisms $\mathcal{M}_1,\ldots,\mathcal{M}_k$, for any λ:*

$$\alpha_{\mathcal{A}}(\lambda) \le \sum_{i=1}^{k} \alpha_{\mathcal{M}_i}(\lambda) \tag{2.7}$$

2. ***Tail Bound:*** *For any $\epsilon > 0$ and λ, the stochastic algorithm, $\mathcal{A}$, is (ϵ,δ)-differentially private for*

$$\delta \le \exp(\alpha_{\mathcal{A}}(\lambda) - \lambda\epsilon) \tag{2.8}$$

The moments accountant scheme computes $\alpha_\alpha(\mathcal{M}_i)$ for each step for several values of λ and uses the composability property to compute the aggregate loss. [8] In typical applications, either the target value of ϵ or δ is fixed at some value, and thus the tail bound property is applied to compute the best possible value of the other privacy variable, using the appropriate following equation:

$$\delta = \min_{\lambda} \exp(\alpha_{\mathcal{A}}(\lambda) - \lambda\epsilon) \tag{2.9}$$

$$\epsilon = \min_{\lambda} \frac{1}{\lambda}(\alpha_{\mathcal{A}}(\lambda) - \ln\delta) \tag{2.10}$$

Let $\mathcal{D} = \{\boldsymbol{x}_i\}$ and $\mathcal{D}' = \mathcal{D} \cup \boldsymbol{x}'$ where $\boldsymbol{x} \in \mathcal{X}$. Let $f : \mathcal{X} \to \mathbb{R}^n$ with $||f(\cdot)||_2 \le \Delta$. Consider the following mechanism, known as the *Gaussian Mechanism with sub-sampling*

$$\mathcal{M}(\mathcal{D}) = \sum_{i\in J} f(\boldsymbol{x}_i) + \mathcal{N}(\boldsymbol{0}, \Delta^2\sigma^2\boldsymbol{I}) \tag{2.11}$$

where i is a subset of indices where each index is chosen independently with probability q. Without loss of generality, let $f(\boldsymbol{x}_i) = \boldsymbol{0}$ and $f(\boldsymbol{x}') = \Delta \cdot \boldsymbol{e}$ where $\boldsymbol{e}_1$ is a unit vector. Then $\mathcal{M}(\mathcal{D})$ and $\mathcal{M}(\mathcal{D}')$ are distributed identically other than the coordinate corresponding to $\boldsymbol{e}_1$. Considering only this direction, the output densities for the mechanism are:

$$\mathcal{M}(\mathcal{D}) \sim \mu_0 \triangleq \mathcal{N}(0, \Delta^2\sigma^2) \tag{2.12}$$
$$\mathcal{M}(\mathcal{D})' \sim \mu_1 \triangleq q \cdot \mathcal{N}(\Delta, \Delta^2\sigma^2) + (1-q) \cdot \mathcal{N}(0, \Delta^2\sigma^2) \tag{2.13}$$

Then, $\alpha_{\mathcal{M}}(\lambda)$ can be computed as:

$$\alpha_{\mathcal{M}}(\lambda) = \ln \max(E_1, E_2) \tag{2.14}$$
$$E_1 = \mathbb{E}_{z\sim\mu_0}\left[\left(\frac{\mu_0(z)}{\mu_1(z)}\right)^{\lambda}\right] \tag{2.15}$$
$$E_2 = \mathbb{E}_{z\sim\mu_1}\left[\left(\frac{\mu_1(z)}{\mu_0(z)}\right)^{\lambda}\right] \tag{2.16}$$

where each integral can be evaluated using numerical integration. Both integrals must be considered since the maximum of the log moments of the privacy loss must be found, and both $\mathcal{D}$ and $\mathcal{D}'$ could be considered as the 'original' dataset. [8]

Note: the computed values of ϵ and δ are independent of the data used for the mechanism and could be pre-computed given fixed values of Δ, q and either δ or ϵ.

2.1.3 Differentially Private Stochastic Gradient Descent

Abadi et. al propose differentially private stochastic gradient descent by adapting stochastic gradient descent to use the Gaussian mechanism with sub-sampling to compute the gradient of some loss function. [8]

Algorithm 1 outlines the differentially private stochastic gradient descent method. Note that the gradient clipping used in this method effectively limits the contribution of each data-points towards the gradient, bounding the ℓ_2 sensitivity and enabling the Gaussian mechanism with sub-sampling to be applied. Noting that the noise added scales with the clipping bound, large values of the clipping bound correspond to strong, but noisy gradient signals. Abadi et. al suggest setting c_t to be fixed at the median of the norms of unclipped gradients throughout the course of training as well as using a learning rate with starts off relatively large and is reduced over time. Additionally, it is remarked that $L \simeq \sqrt{N}$ is appropriate; larger values of L improve the accuracy of the gradient signal but increase the privacy cost. [8]

ms2314: Haven't mentioned privacy amplification - perhaps I outhg to here

In Abadi et al. (2016), this technique was applied to deep neural networks. However, this algorithm can be used as a building block to create other differentially private techniques,

Algorithm 1 Differentially Private Stochastic Gradient Descent (DP-SGD)

```
Input: Dataset $\mathcal{D} = \{\boldsymbol{x}_1, \dots, \boldsymbol{x}_N\}$, Loss function $\mathcal{L}(\boldsymbol{\theta}) = \frac{1}{N}\sum_i \mathcal{L}(\boldsymbol{\theta}, \boldsymbol{x}_i)$
Parameters: Learning Rate $\eta_t$, Clipping Bound $c_t$, Lot Size $L$, DP Noise Scale $\sigma_t$,
Num. Iterations $T$
1: Initialise $\boldsymbol{\theta}_0$ randomly.
2: for $t = 1, \dots, T$ do
3:     Take a random sample $L_t$ with sampling probability $q = L/N$
4:     for all $i \in L_t$ do                                          ▷ Compute Gradient
5:         $\boldsymbol{g}_t(\boldsymbol{x}_i) \leftarrow \nabla_{\boldsymbol{\theta}} \mathcal{L}(\boldsymbol{\theta}, \boldsymbol{x}_i)$
6:         $\tilde{\boldsymbol{g}}_t(\boldsymbol{x}_i) \leftarrow \boldsymbol{g}_t(\boldsymbol{x}_i) / \max(||\boldsymbol{g}_t(\boldsymbol{x}_i)||_2 / c_t, 1)$      ▷ Gradient Clipping
7:     end for
8:     $\tilde{\boldsymbol{g}}_t \leftarrow \frac{1}{L}[\sum_{i \in L_t} \boldsymbol{g}_t(\boldsymbol{x}_i) + \mathcal{N}(0, \sigma_t^2 c_t^2 \boldsymbol{I})]$      ▷ Perturb Clipped Gradient
9:     $\boldsymbol{\theta}_t \leftarrow \boldsymbol{\theta}_{t-1} - \eta_t \tilde{\boldsymbol{g}}_t$
10: end for
11: Output $\boldsymbol{\theta}_T$ and calculate $(\epsilon, \delta)$ using the Moments Accountant.
```

and indeed this technique has been applied in order to perform variational inference for non conjugate models [9].

2.2 Federated Learning

2.2.1 Partitioned Variational Inference

Partitioned Variational Inference (PVI) is a general framework which encompasses many variational Bayesian techniques. Assume that the dataset, $\mathcal{D}$ has been partitioned into M shards i.e. $\mathcal{D} = \{\boldsymbol{X}_1, \dots, \boldsymbol{X}_M\}$, where $\boldsymbol{X}_i = \{\boldsymbol{x}_1, \dots, \boldsymbol{x}_{N_i}\}$. A probabilistic model with parameters $\boldsymbol{\theta}$ has been suggested to model this data with a known prior, $p(\boldsymbol{\theta})$ and likelihood function, $p(\boldsymbol{x}|\boldsymbol{\theta})$. The aim of Bayesian inference is to calculate the posterior density over the parameters, $p(\boldsymbol{\theta}|\mathcal{D})$ but in general, it is not possible to compute this distribution. In variational methods, an approximate distribution, $q(\theta)$, is used to approximate the posterior. In PVI, the proposal distribution takes the form:

$$q(\boldsymbol{\theta}) = p(\boldsymbol{\theta}) \prod_{m=1}^{M} t_m(\boldsymbol{\theta}) \simeq \frac{1}{\mathcal{Z}} p(\boldsymbol{\theta}) \prod_{m=1}^{M} p(\boldsymbol{X}_m|\boldsymbol{\theta}) = p(\boldsymbol{\theta}|\mathcal{D}) \tag{2.17}$$

Inspecting the above equation, it can be seen that each $t_m(\boldsymbol{\theta})$ approximates the (un-normalised) likelihood $p(\boldsymbol{X}_m|\boldsymbol{\theta})$, noting that the approximate posterior does not include a normalising constant.

When partitioned variational inference is used in the federated learning context, each client stores its own data partition, $\boldsymbol{X}_m$, and communicated with a central parameter server which stores the global approximate posterior. At each stage, each client maximises a *local* free energy. [10]

Algorithm 2 Partitioned Variational Inference (PVI)

```
Input: Partitioned Dataset $\mathcal{D} = \{\boldsymbol{X}_1, \dots, \boldsymbol{X}_M\}$, Prior $p(\boldsymbol{\theta})$, Family of Distributions $\mathcal{Q}$
1: Initialise approximate likelihood:
       $$t_m^{(0)}(\boldsymbol{\theta}) \leftarrow 1 \ \forall m \quad (2.18)$$
2: Initialise approximate posterior:
       $$q^{(0)}(\boldsymbol{\theta}) \leftarrow p(\boldsymbol{\theta}) \quad (2.19)$$
3: for $i = 1, 2, \dots,$ until convergence do
4:     $b_i \leftarrow$ index of next approximate likelihood to update.
5:     Compute the new approximate likelihood:
       $$q^{(i)}(\boldsymbol{\theta}) \leftarrow \underset{q(\boldsymbol{\theta}) \in \mathcal{Q}}{\operatorname{argmax}} \int q(\boldsymbol{\theta}) \ln \frac{q^{(i-1)}(\boldsymbol{\theta}) p(\boldsymbol{X}_{b_i}|\boldsymbol{\theta})}{q(\boldsymbol{\theta}) t_{b_i}^{(i-1)}(\boldsymbol{\theta})} \, d\boldsymbol{\theta} \quad (2.20)$$
6:     Update the approximate likelihood for the updated factor:
       $$t_{b_i}^{(i)}(\boldsymbol{\theta}) \leftarrow \frac{q^{(i)}(\boldsymbol{\theta})}{q^{(i-1)}(\boldsymbol{\theta})} t_{b_i}^{(i-1)}(\boldsymbol{\theta}) \quad (2.21)$$
7: end for
```

Definition 2.2.1 (Local Free Energy) *The local free energy is defined as:*

$$\mathcal{F}_{b_i}^{(i)}(q(\boldsymbol{\theta})) = \int q(\boldsymbol{\theta}) \ln \frac{q^{(i-1)}(\boldsymbol{\theta}) p(\boldsymbol{X}_{b_i}|\boldsymbol{\theta})}{q(\boldsymbol{\theta}) t_{b_i}^{(i-1)}(\boldsymbol{\theta})} \, d\boldsymbol{\theta} \quad (2.22)$$

It can be shown that the local free energy optimisation (Eq. 2.20) is equivalent to the following $\mathcal{KL}$ optimisation:

$$q^{(i)}(\boldsymbol{\theta}) \leftarrow \underset{q(\boldsymbol{\theta}) \in \mathcal{Q}}{\operatorname{argmax}} \mathcal{KL}(q(\boldsymbol{\theta}) || \hat{p}_{b_i}^{(i)}(\boldsymbol{\theta})) \quad (2.23)$$

where $\hat{p}_{b_i}^{(i)}(\boldsymbol{\theta})$ is known as the *tilted distribution* and is defined as:

$$\begin{aligned} \hat{p}_{b_i}^{(i)}(\boldsymbol{\theta}) &= \frac{1}{\mathcal{Z}_{b_i}^{(i)}} \frac{q^{(i-1)}(\boldsymbol{\theta})}{t_{b_i}^{(i-1)}(\boldsymbol{\theta})} p(\boldsymbol{X}_{b_i}|\boldsymbol{\theta}) \quad (2.24) \\ &= \frac{1}{\mathcal{Z}_{b_i}^{(i)}} p(\boldsymbol{\theta}) p(\boldsymbol{X}_{b_i}|\boldsymbol{\theta}) \prod_{m \neq b_i} t_m^{(i-1)}(\boldsymbol{\theta}) \quad (2.25) \end{aligned}$$

which can be interpreted as a local estimate of the posterior for client b_i, using the exact local likelihood and approximate likelihood terms for the data partitions held at other clients.

In standard variational inference, a global free energy, depending on the entire dataset is maximised.

Definition 2.2.2 (Global Free Energy) *The global free energy is defined as:*

$$\mathcal{F}(q(\boldsymbol{\theta})) = \int q(\boldsymbol{\theta}) \ln \frac{p(\boldsymbol{\theta}) \prod_{m=1}^{M} p(\boldsymbol{X}_m|\boldsymbol{\theta})}{q(\boldsymbol{\theta})} \, d\boldsymbol{\theta} \tag{2.26}$$

Maximising this free energy is equivalent to minimising the $\mathcal{KL}$ divergence between the approximate posterior and the true posterior. [11]

Now we return to the partitioned variational inference setting.

Theorem 2.2.3 (Properties of PVI) *Consider the approximate posterior at convergence* $q^*(\boldsymbol{\theta}) = p(\boldsymbol{\theta}) \prod_{m=1}^{M} t_m^*(\boldsymbol{\theta})$*. Then:*

1. *The sum of local free energies is the global free energy:*

$$\sum_{m=1}^{M} \mathcal{F}_m(q^*(\boldsymbol{\theta})) = \mathcal{F}(q^*(\boldsymbol{\theta})) \tag{2.27}$$

2. *Optimising the local free energies optimises the global free energy:*

$$q^*(\theta) = \underset{q(\boldsymbol{\theta}) \in \mathcal{Q}}{\operatorname{argmax}} \mathcal{F}_m(q(\boldsymbol{\theta})) \ \forall m \Rightarrow q^*(\theta) = \underset{q(\boldsymbol{\theta}) \in \mathcal{Q}}{\operatorname{argmax}} \mathcal{F}(q(\boldsymbol{\theta})) \tag{2.28}$$

The above properties of PVI motivate the scheme and suggest that it may provide reasonable result. [10]

ms2314: perhaps remove this, may have gone into too much depth here.

Chapter 3

Design of Distributed DP Mechanisms

Chapter 4

Results & Discussion

Chapter 5

Conclusions

References

[1] Teppo Niinimäki, Mikko Heikkilä, Antti Honkela, and Samuel Kaski. Representation transfer for differentially private drug sensitivity prediction. *arXiv preprint arXiv:1901.10227*, 2019.

[2] A. Narayanan and V. Shmatikov. Robust de-anonymization of large sparse datasets. In *2008 IEEE Symposium on Security and Privacy (sp 2008)*, pages 111–125, May 2008. doi: 10.1109/SP.2008.33.

[3] Melissa Gymrek, Amy L McGuire, David Golan, Eran Halperin, and Yaniv Erlich. Identifying personal genomes by surname inference. *Science*, 339(6117):321–324, 2013.

[4] Matt Fredrikson, Somesh Jha, and Thomas Ristenpart. Model inversion attacks that exploit confidence information and basic countermeasures. In *Proceedings of the 22Nd ACM SIGSAC Conference on Computer and Communications Security*, CCS '15, pages 1322–1333, New York, NY, USA, 2015. ACM. ISBN 978-1-4503-3832-5. doi: 10.1145/2810103.2813677. URL http://doi.acm.org/10.1145/2810103.2813677.

[5] Federated learning: Collaborative machine learning without centralized training data, Apr 2017. URL https://ai.googleblog.com/2017/04/federated-learning-collaborative.html.

[6] Cynthia Dwork and Aaron Roth. The algorithmic foundations of differential privacy. *Found. Trends Theor. Comput. Sci.*, 9(3&#8211;4):211–407, August 2014. ISSN 1551-305X. doi: 10.1561/0400000042. URL http://dx.doi.org/10.1561/0400000042.

[7] Sebastian Ruder. An overview of gradient descent optimization algorithms. *arXiv preprint arXiv:1609.04747*, 2016.

[8] Martin Abadi, Andy Chu, Ian Goodfellow, H. Brendan McMahan, Ilya Mironov, Kunal Talwar, and Li Zhang. Deep learning with differential privacy. In *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*, CCS '16, pages 308–318, New York, NY, USA, 2016. ACM. ISBN 978-1-4503-4139-4. doi: 10.1145/2976749.2978318. URL http://doi.acm.org/10.1145/2976749.2978318.

[9] J. Jälkö, O. Dikmen, and A. Honkela. Differentially Private Variational Inference for Non-conjugate Models. *ArXiv e-prints*, October 2016.

[10] Thang D Bui, Cuong V Nguyen, Siddharth Swaroop, and Richard E Turner. Partitioned variational inference: A unified framework encompassing federated and continual learning. *arXiv preprint arXiv:1811.11206*, 2018.

[11] Christopher M. Bishop. *Pattern Recognition and Machine Learning*. Springer, 2006. ISBN 978-0387-31073-2. URL http://research.microsoft.com/en-us/um/people/cmbishop/prml/.