Representation Learning and Applications in Local Differential Privacy

Alexander Mansbridge

Department of Computer Science

University College London

This dissertation is submitted for the degree of

Doctor of Philosophy

Declaration

I, Alexander Mansbridge, confirm that the work presented in this thesis is my own. Where information has been derived from other sources, I confirm that this has been indicated in the thesis.

Abstract

Latent variable models (LVMs) provide an elegant, efficient, and interpretable approach to learning the generation process of observed data. Latent variables can capture salient features within often highly-correlated data, forming powerful tools in machine learning.

For high-dimensional data, LVMs are typically parameterised by deep neural networks, and trained by maximising a variational lower bound on the data log likelihood. These models often suffer from poor use of their latent variable, with ad-hoc annealing factors used to encourage retention of information in the latent variable. In this work, we first introduce a novel approach to latent variable modelling, based on an objective that encourages both data reconstruction and generation. This ensures by design that the latent representations capture information about the data.

Second, we consider a novel approach to inducing local differential privacy (LDP) in high dimensions with a specifically-designed LVM. LDP offers a rigorous approach to preserving one's privacy against both adversaries and the database administrator. Existing LDP mechanisms struggle to retain data utility in high dimensions owing to prohibitive noise requirements. We circumvent this by inducing LDP on the low-dimensional manifold underlying the data. Further, we introduce a novel approach for downstream model learning using LDP training data, enabling the training of performant machine learning models. We achieve significant performance gains over current state-of-the-art LDP mechanisms, demonstrating far-reaching implications for the widespread practice of data collection and sharing.

Finally, we scale up this approach, adapting current state-of-the-art representation learning models to induce LDP in even higher-dimensions, further widening the scope of LDP mechanisms for high-dimensional data collection.

Impact statement

The research in this thesis presents potential benefits both inside and outside of academia. Notably, Chapters 4 and 5 highlight the benefits of using representation learning for privatising high-dimensional data under local differential privacy (LDP). Not only do we significantly outperform current state-of-the-art LDP mechanisms, but we also present a number of avenues of future academic research that could further improve the performance of such LDP mechanisms. Indeed, we see this work as a significant first step for LDP representation learning.

These mechanisms could also have far-reaching implications outside academia, where data collection practices have become ubiquitous, and often pervasive. The framework we introduce is intuitive, general, and straightforward to implement. Existing LDP mechanisms are limited in application since they significantly degrade data utility in high-dimensions. However, having demonstrated that high-dimensional privatisation with our mechanisms is not only feasible, but can lead to compelling downstream model performance, we hope that this should encourage data-collecting organisations to consider collecting under LDP guarantees.

The work in Chapter 3 has contributed to our understanding of latent variable models (LVMs). We introduce a novel LVM that encodes more information into the latent than the standard variational autoencoder (VAE), and provides a principled interpretation for the use of ad-hoc pre-factors in VAE objective functions, common in existing academic work. This has potential implications both in academia and in industry, where interest in LVMs has grown steadily in recent years, owing to the widespread availability of unlabelled training data.

The work presented in this thesis has been published in a major machine learning conference [Mansbridge et al., 2019].

Acknowledgements

I would like to express my gratitude to Prof. David Barber and Dr. Ilya Feige for their invaluable supervision and feedback throughout this work, and to the Alan Turing Institute for their academic, financial and pastoral support. Thanks to my examiners Dr. Tingting Mu and Dr. Brooks Paige for a thorough, enjoyable viva and outstanding feedback. Finally, a special thank you to my family for their unwavering love and support, without whom I would undoubtedly never have had the privilege of embarking on the research in this thesis to begin with.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

In machine learning, and indeed science more broadly, a common goal is to build models of the world. In probabilistic learning we typically consider the world, which we observe through data, as being described by a set of variables. Modelling the distributions over these variables provides us with a powerful framework for dealing with uncertainty in this world. It allows us to make informed predictions; to do inference, answering pertinent questions about unobserved variables; and to understand the underlying structure of the data-generating process.

Probabilistic generative models have been used in statistics for decades, aiming to simulate the process by which data is generated. Over the last few years however, rapid advancements in machine learning research have given rise to a new generation of extremely powerful generative models. These models are the product of myriad advancements in the field, notably benefitting from the use of deep neural network architectures to parameterise distributions, wider access to powerful compute resources, greater availability of large scale datasets, and ever-improving algorithm design. A key advantage of generative models is that they are often trained in an unsupervised manner, presenting a powerful way to leverage vast quantities of freely available, unlabelled data. This has facilitated the training of large models that act on high-dimensional data with compelling results.

Generative models can be broadly classified into two categories: fully observed models and latent variable models.

Fully observed models, such as Salimans et al. [2017], often utilise powerful,

auto-regressive architectures to achieve state-of-the-art likelihood performance, making them excellent candidates for generating realistic data samples, as well as in applications like data compression.

Latent variable models on the other hand assume our observed data is generated by some random unobserved stochastic variable(s). They provide an interpretable, efficient approach to modelling the data generation process.

State-of-the-art latent variable models are typically based on the variational autoencoder (VAE) framework, trained by maximising a variational lower bound on the data log likelihood [Kingma and Welling, 2014, Rezende et al., 2014]. VAEs learn a generative distribution, and fit an approximate posterior to the intractable posterior distribution concurrently, utilising deep neural networks to parameterise complex, non-linear distributions. The approximate posterior parameterisation is chosen such that we can perform approximate maximum likelihood estimation using Stochastic Gradient Variational Bayes (SGVB).

Models based on VAEs have been used extensively in machine learning, demonstrating compelling results in data generation for images [Gulrajani et al., 2017, Child, 2021], text [Yang et al., 2017, Shah et al., 2018], and speech [van den Oord et al., 2017, Lee et al., 2021], as well as in widespread applications such as semi-supervised learning [Kingma et al., 2014, Habib et al., 2020], data compression [Townsend et al., 2019, Kingma et al., 2019] and explainable AI [Frye et al., 2021]. However, it is often the case that, while the model is able to maximise a lower bound on the data log likelihood, further modifications are needed to ensure the model utilises the latent variable [Bowman et al., 2016, Higgins et al., 2017, He et al., 2019]. In this work, we study latent variable models with a particular focus on encoding meaningful information into the latent.

A widespread practice in machine learning is to use bigger neural networks as the data dimension grows, or the data distribution becomes more complex. However, when training VAEs with more expressive generative distributions parameterised by larger neural networks, a phenomenon known as posterior collapse often occurs. In this scenario, little to no information is encoded into the latent variable, and the generative model learns to ignore it. Several workarounds have been proposed,

perhaps most commonly the use of ad-hoc annealing factors in the objective function (see, for example Bowman et al. [2016]). However, this raises a compelling argument that there may exist better objective functions for training latent variable models that necessitate the encoding of information in the latent variable.

In Chapter 3, we introduce an alternative and general approach to latent variable modelling, based on an objective that encourages both data generation and reconstruction. This ensures by design that the latent variable captures information about the observations, whilst retaining the ability to generate well. Interestingly, although our model is fundamentally different to a VAE, the lower bound attained is identical to the standard VAE bound but with a simple pre-factor, thus providing a formal interpretation of the ad-hoc pre-factors commonly used when training VAEs. We demonstrate the effectiveness of our approach with language modelling. Language represents a complex, high-dimensional, and highly sparse data type. Given the powerful network architectures typically used to model such data, optimisation challenges associated with utilisation of the latent variable in standard VAEs are well documented [Bowman et al., 2016, Yang et al., 2017].

In Chapter 4, we develop a latent variable model designed specifically for data collection under local differential privacy (LDP). In recent years, the collection of personal data has become ubiquitous, with collection practices often considered invasive, or even a violation of human rights [Amnesty International, 2019]. As awareness of these often contentious practices has grown, both companies' desire to surveil individuals, and individuals' desire for data privacy have firmly entered into the modern zeitgeist.

LDP is a rigorous privacy guarantee that protects an individuals' data against both the database administrator and adversarial third parties, providing a natural framework for privacy protection in the context of data collection. While mechanisms that privatise data under the guarantees of LDP were first introduced decades ago [Warner, 1965], research into LDP mechanisms for the high-dimensional data coveted by many modern organisations remains limited. Indeed, research shows that the quantity of noise required to induce privacy in higher dimensions typically destroys data utility in what is commonly dubbed the 'curse of dimensionality' [Zhang et al.,

2017, Duchi et al., 2018].

A common motivating assumption in latent variable modelling is that our high-dimensional data lies on a low-dimensional manifold. Our model defines a LDP mechanism via a learnt distribution over this manifold. This distribution has a constrained mean, such that adding carefully calibrated noise guarantees both the latent encoding and reconstructed data satisfy LDP. Passing data through this latent variable model therefore provides a powerful LDP mechanism for the private collection of sensitive, high-dimensional data. Through privatisation on this constrained, low-dimensional manifold, we circumvent the LDP-inducing noise requirements in high-dimensional data space. We train our latent variable model such that the latent variables are well-suited for downstream tasks, suitably constrained for privatisation, and robust to the additive noise that induces LDP.

While Chapter 4 introduces a powerful framework for learning LDP latent representations of data, we note that the performance of this mechanism is fundamentally limited by the quantity of information encoded into the latent. As discussed in Chapter 3, when modelling very high-dimensional data, there are challenges associated with training latent variable models such that information is encoded into the latent. The performance of the LDP mechanism from Chapter 4 will thus likely be impacted. We also note that if we are solely interested in representation quality, learning the mapping from representation space to feature space is unnecessary, making optimisation more difficult and adding computational cost. In Chapter 5, we propose a solution to tackle these challenges, building on ideas from the much wider field of representation learning in order to privatise even higher-dimensional data.

Modern representation learning methods can be broadly split into generative approaches, such as those discussed in Chapters 3 and 4, and non-generative approaches. Non-generative approaches typically learn only the mapping from data space to representation space, with unsupervised, and more recently self-supervised approaches achieving state-of-the-art results across high-dimensional domains like large images [Chen et al., 2020, He et al., 2020] and text [Devlin et al., 2019, Yang et al., 2021]. Chapter 5 introduces a clear framework for adapting existing state-of-the-art non-generative representation learning models to the application of LDP

data collection. We demonstrate this approach empirically by adapting the self-supervised approach introduced by Chen et al. [2020] to learn powerful, noise-robust representations of colour images.

Throughout Chapters 4 and 5, we use our privatised datapoints as training data for learning downstream machine learning algorithms. This is not only a key goal of data collecting organisations in the real world, but downstream model performance provides a powerful proxy for measuring the utility of our privatised datapoints. To accomplish this, we introduce a downstream model in which our observed data is privatised, whilst the true underlying data and targets are treated as latent variables. We aim to learn the mapping between these latents. We demonstrate empirically that models trained on data privatised with our mechanism significantly outperform models trained on data privatised with existing state-of-the-art LDP mechanisms.

Finally, in Chapter 6, we discuss some general conclusions of the research in this thesis and discuss several potential directions for future work. However, before introducing the main research contributions of this thesis, we begin with a background chapter, outlining the fundamental concepts considered prerequisite to this thesis. This includes an overview of probabilistic generative models, an introduction to recurrent neural networks and related concepts, and an introduction to key definitions and concepts in differential privacy.

Chapter 2

Background

This chapter introduces background material required to understand the thesis. We include both a concise overview of the overarching concepts, as well as definitions which we refer to throughout the following chapters. We begin with an overview of probabilistic generative models in Section 2.1, with a particular focus on latent variable models. We discuss training such models with both the EM algorithm and stochastic gradient variational Bayes. In Section 2.2, we give an overview of neural networks and some related concepts discussed in Chapter 3. In Section 2.3, we motivate the need for data privacy, and introduce the concepts of central differential privacy and local differential privacy, as well as the fundamental differences between these two privacy models.

2.1 Probabilistic Generative models

Generative models are a class of statistical model that aim to simulate the process by which observed datapoints are generated. We can think of these datapoints as being observed samples from some unknown distribution $p_{\text{data}}(\cdot)$. Naturally, we would like to learn a generative model that closely matches this distribution. In this work, we consider the parametric setting – we assume this process is defined by some distribution $p_\theta(\cdot)$ with (deterministic) parameters θ. In order to learn these parameters, we must minimise some measure of distance between the unknown data generating distribution $p_{\text{data}}(\cdot)$ and the model $p_\theta(\cdot)$. Throughout this work, we learn

these parameters with maximum likelihood estimation. That is, we want to find the parameters $\hat{\theta}$ such that

$$\hat{\theta} = \arg\max_{\theta} \mathbb{E}_{x \sim p_{\text{data}}(x)} \big[\log p_\theta(x) \big] \tag{2.1}$$

We note that this is equivalent to minimising the Kullback–Leibler (KL) divergence between the data generating distribution $p_{\text{data}}(\cdot)$ and the model $p_\theta(\cdot)$:

$$\hat{\theta} = \arg\min_{\theta} D_{\text{KL}} \big[p_{\text{data}}(x) || p_\theta(x) \big] \tag{2.2}$$

where for continuous distributions $p_1(\cdot)$ and $p_2(\cdot)$ over $\mathcal{X}$, the KL divergence is defined as follows:

$$D_{\text{KL}} \big[p_1(x) || p_2(x) \big] = \int_{\mathcal{X}} p_1(x) \log \frac{p_1(x)}{p_2(x)} dx \tag{2.3}$$

2.1.1 Latent Variable Models

It is often good to assume that the data we observe lies on some lower-dimensional manifold. Latent variable models (LVMs) are a class of generative model that assume the data is generated by the transformation of some underlying latent variable(s). This dependency structure can lead to richer, more expressive models where often complex correlations in the observed data x are modelled through dependencies on the latent z. Furthermore, the low-dimensional latent can reveal properties of our data that are more interpretable and more suitable for downstream applications than the observed data.

If the latent variable were observed, learning would be straightforward since we could just maximise the joint likelihood. However, since this is not the case, we instead maximise the log likelihood of our generative model by marginalising out z:

$$p_\theta(x) = \int p_\theta(x|z)\, p_\theta(z)\, dz \tag{2.4}$$

where $p_\theta(z)$ describes our prior belief over the latent space, and $p_\theta(x|z)$ describes the generative model we wish to learn. In many instances $p_\theta(z)$ and $p_\theta(x|z)$ may be from exponential family distributions, but $p_\theta(x)$ is able to model far richer distributions

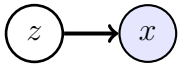


Figure 2.1: A graphical model representing the dependency structure for a latent variable model.

than a simple exponential family model.

Figure 2.1 shows a directed graphical model for the latent variable model described in Equation 2.4. Graphical models provide a simple yet powerful way to illustrate the dependency structure between variables in probabilistic models. Each node represents a random variable, and directed edges between nodes indicate statistical dependency between the corresponding variables. The blue shading indicates that the random variable is observed.

Often direct maximum likelihood optimisation of Equation 2.4 is not straightforward. Instead, we can derive a lower bound on the integral using Jensen's inequality:

$$\begin{aligned}
\log p_\theta(x) &= \log \int p_\theta(x|z)p_\theta(z)\,dz && (2.5)\\
&= \log \int \frac{q(z)}{q(z)} p_\theta(x|z)p_\theta(z)\,dz && (2.6)\\
&\geq \int q(z) \log \frac{p_\theta(x|z)p_\theta(z)}{q(z)}\,dz && (2.7)\\
&= \int q(z) \log p_\theta(x|z) dz - \int q(z) \log \frac{q(z)}{p_\theta(z)}\,dz && (2.8)\\
&= \mathbb{E}_{q(z)}\big[\log p_\theta(x|z)\big] - D_{\mathrm{KL}}\big[q(z)||p_\theta(z)\big] && (2.9)\\
&=: \mathcal{L}(x; q, \theta) && (2.10)
\end{aligned}$$

where we have introduced a variational distribution $q(z)$. This lower bound $\mathcal{L}(x; q, \theta)$ is commonly referred to in the literature as the Evidence Lower Bound (ELBO).

It will be useful in the next section to note the following property of this lower bound:

$$\begin{aligned}
\mathcal{L}(x; q, \theta) &= \int q(z) \log \frac{p_\theta(x)p_\theta(z|x)}{q(z)}\,dz && (2.11)\\
&= \log p_\theta(x) + \int q(z) \log \frac{p_\theta(z|x)}{q(z)}\,dz && (2.12)
\end{aligned}$$

$$= \log p_\theta(x) - D_{\mathrm{KL}}\big[q(z)||p_\theta(z|x)\big] \tag{2.13}$$

This states that the difference between the ELBO and the true log likelihood is equal to $D_{\mathrm{KL}}\big[q(z)||p_\theta(z|x)\big]$. Since the (non-negative) KL is minimised if and only if the two distributions are identical, it follows that the ELBO is equal to the log likelihood only when $q(z)$ is equal to the posterior $p_\theta(z|x)$.

2.1.1.1 The EM Algorithm

The EM algorithm [Dempster et al., 1977] provides a powerful, iterative algorithm for optimising the log likelihood of a latent variable model using this lower bound. Starting from some arbitrary choice of $q(z)$ and θ, we iterate over the following two steps:

- **E step:** Fix the parameters θ of the generative model and update the variational distribution

$$q(z) := \arg\max_q \mathcal{L}(x; q, \theta) \tag{2.14}$$

- **M step:** Fix the variational distribution $q(z)$ and update the parameters θ of the generative distribution

$$\theta := \arg\max_\theta \mathcal{L}(x; q, \theta) \tag{2.15}$$

Examining Equation 2.13, we see that we can maximise the E-step by setting $q(z) := p_\theta(z|x)$ (assuming the true posterior can be explicitly calculated).

2.1.1.2 Variational Autoencoders

In order to increase the representational power of the latent variable model, a deep neural network may be used to parameterise $p_\theta(x|z)$. This non-linear conditional dependency on the latent means the integral in Equation 2.4 is generally intractable; we must therefore optimise the ELBO rather than directly optimising the likelihood. However, we cannot use the EM algorithm as described above since the true posterior will not generally have a known closed-form solution.

Instead, much like the generative distribution $p_\theta(x|z)$, we parameterise the variational distribution $q_\phi(z|x)$ with a neural network. This amortises the cost of inference by sharing the network parameterising the variational distribution across datapoints.

Both the variational and generative distributions can be trained by maximising a lower bound on the log likelihood:

$$\log p_\theta(x) = \log \int p_\theta(x|z)p(z)\,dz \tag{2.16}$$

$$\geq \mathbb{E}_{q_\phi(z|x)}\big[\log p_\theta(x|z)\big] - D_{\mathrm{KL}}\big[q_\phi(z|x)||p(z)\big] \tag{2.17}$$

$$=: \mathcal{L}(x;\theta,\phi) \tag{2.18}$$

Taking gradients of $\mathcal{L}(x;\theta,\phi)$ with respect to ϕ can be challenging. The following Monte Carlo estimate of the first term can be used:

$$\nabla_\phi \mathbb{E}_{q_\phi(z|x)}\big[\log p_\theta(x|z)\big] = \int \nabla_\phi q_\phi(z|x)\log p_\theta(x|z)\,dz \tag{2.19}$$

$$= \int \big[q_\phi(z|x)\nabla_\phi \log q_\phi(z|x)\big]\log p_\theta(x|z)\,dz \tag{2.20}$$

$$\simeq \frac{1}{S}\sum_{s=1}^{S}\big(\nabla_\phi \log q_\phi(z^{(s)}|x)\big)\log p_\theta(x|z^{(s)}) \tag{2.21}$$

where $z^{(s)} \sim q_\phi(z|x)$.

However, the variance of this Monte Carlo estimate can be extremely high, as discussed in Paisley et al. [2012]. To circumvent this, Kingma and Welling [2014] and Rezende et al. [2014] introduced an unbiased estimator of the ELBO, known as the Stochastic Gradient Variational Bayes (SGVB) estimator. This places some restrictions on our choice of variational distribution: namely that the random variable $z \sim q_\phi(z|x)$ must be a reparameterisation of some other variable $\epsilon \sim p(\epsilon)$, via some differentiable transformation $z = g_\phi(\epsilon, x)$. Then we see that $\mathbb{E}_{z\sim q(z|x)}\big[f(z)\big] = \mathbb{E}_{\epsilon\sim p(\epsilon)}\big[f(g_\phi(\epsilon,x))\big]$ and gradients can be estimated as follows

$$\nabla_\phi \mathbb{E}_{z\sim q_\phi(z|x)}\big[f(z)\big] = \nabla_\phi \mathbb{E}_{\epsilon\sim p(\epsilon)}\big[f(g_\phi(\epsilon,x))\big] \tag{2.22}$$

$$= \mathbb{E}_{\epsilon\sim p(\epsilon)}\big[\nabla_\phi f(g_\phi(\epsilon,x))\big] \tag{2.23}$$

$$\simeq \frac{1}{S}\sum_{s=1}^{S} \nabla_\phi f(g_\phi(\epsilon^{(s)}, x)) \tag{2.24}$$

where $\epsilon^{(s)} \sim p(\epsilon)$.

This is commonly referred to as the 'reparameterisation trick'. We note that this property holds for both Gaussian and Laplace distributions, which we use in this work. For example, if can sample $\epsilon \sim \mathbb{N}(0, \mathbb{I})$ then $z = \mu_\phi(x) + \sigma_\phi(x) \odot \epsilon$ is distributed according to $\mathbb{N}(\mu_\phi(x), \sigma_\phi(x))$. For further details on the reparameterisation trick, and a (non-exhaustive) list of distributions for which this property holds we refer the reader to Kingma and Welling [2014].

2.2 Neural Network Architectures

Deep learning is a subfield of machine learning in which algorithms based on neural networks are trained to act as powerful function approximators; it can be shown that under certain conditions neural networks are universal approximators [Hornik, 1991], making them powerful tools in machine learning.

A neural network is an interconnected collection of neurons – a computational unit that performs non-linear transformations on input vectors. Each neuron is associated with an n-dimensional weight vector w, scalar bias term b, and a non-linear activation function $\sigma(\cdot)$. Given an n-dimensional input vector x, the neuron outputs the transformation $\sigma(w^T x + b)$. Common choices of activation function include the sigmoid, hyperbolic tangent, and rectified linear unit (ReLU) functions [Jarrett et al., 2009, Glorot et al., 2011]. To ease notation, we drop b in the following, observing that it can be absorbed into the dot product as $\sigma(w^T x + b) = \sigma([w \;\; b]^T [x \;\; 1])$.

2.2.1 Feedforward neural networks

Collections of neurons, referred to as layers, form the building blocks of neural networks. These act on vector-valued inputs x to obtain activations h as follows:

$$h = \sigma(W \cdot x) \tag{2.25}$$

where W represents a matrix of parameters.

In deep learning, multiple neural network layers are used to learn powerful function approximators. The simplest example of this is the feedforward neural network, which maps some input datapoint x, to some output h_D:

$$h_0 = x \tag{2.26}$$

$$h_d = \sigma_d(W_d \cdot h_{d-1}), \qquad \text{for } d \in \{1, 2, \cdots, D\} \tag{2.27}$$

Each hidden layer h_d provides some representation of the input data. Typically, the final layer would be used to solve the task of interest. For example, to train a classifier, σ_D could be a softmax layer such that h_D defines a probability distribution over class labels y. Given an appropriate choice of loss function (such as cross-entropy loss for a classifier), one can take gradients with respect to the parameters W_d, and train the network using a gradient based optimisation algorithm [Nesterov, 1983, Duchi et al., 2011, Kingma and Ba, 2015].

2.2.2 Recurrent neural networks

While feedforward networks provide a powerful model for many datatypes, they are not naturally suited to handling sequential data $\{x_1, \ldots, x_T\}$. Recurrent neural networks (RNNs) [Rumelhart et al., 1986] parameterise a time-invariant function $f(\cdot)$ designed to model sequences of variable length. This function maintains and updates a hidden state vector h_t over time

$$h_t = f(h_{t-1}, x_t) \tag{2.28}$$

A simple choice of $f(\cdot)$ may be $f(h_{t-1}, x_t) = \sigma(W \cdot h_{t-1} + W' \cdot x_t)$, where the parameters in the matrices W and W' are shared across timesteps.

In theory, RNNs should be able to capture long term dependencies between datapoints. For example, in language modelling, the task is next word prediction; this prediction may depend not only on the previous word, but on certain words mentioned earlier in the sentence or document. Simple RNNs struggle to learn these

long-term dependencies, and more complex RNN architectures have been introduced to tackle this issue.

Long Short-Term Memory Networks Long Short-term Memory Networks (LSTMs) [Hochreiter and Schmidhuber, 1997] are a powerful class of RNN designed to handle long term dependencies. This is achieved via a memory cell state, which is maintained and controlled over time via three gating functions. There are several variations of the LSTM architecture, but in this thesis, we use the model defined in Graves [2013], performing the following operations at each time step:

$$i_t = \sigma(W_{xi}x_t + W_{hi}h_{t-1} + W_{ci}c_{t-1} + b_i) \tag{2.29}$$

$$f_t = \sigma(W_{xf}x_t + W_{hf}h_{t-1}W_{cf}c_{t-1} + b_f) \tag{2.30}$$

$$c_t = f_t c_{t-1} + i_t \tanh(W_{xc}x_t + W_{hc}h_{t-1} + b_c) \tag{2.31}$$

$$o_t = \sigma(W_{xo}x_t + W_{ho}h_{t-1} + W_{co}c_t + b_o) \tag{2.32}$$

$$h_t = o_t \tanh(c_t) \tag{2.33}$$

where c_t is the cell state at time t; the input gate i_t controls how much new information we encode into c_t; the forget gate f_t controls the extent to which we retain information from c_{t-1}; and the output gate o_t determines the information output to the hidden state h_t at time t.

LSTMs have demonstrated a greater ability to retain longer term dependencies than simple RNNs, with compelling results for language modelling [Graves, 2013, Bowman et al., 2016], machine translation [Sutskever et al., 2014], speech recognition [Graves and Jaitly, 2014], and image captioning [Donahue et al., 2015].

2.2.2.1 Decoding strategies for Recurrent Models

A generative model parameterised by a recurrent network typically factorises the joint likelihood as

$$p(x_1, \ldots, x_T) = p(x_0) \prod_{t=1}^{T} p(x_t | x_0, \ldots, x_{t-1}) \tag{2.34}$$

where at each timestep, the model outputs the distribution $p(x_t|x_0, \ldots, x_{t-1})$ over the current datapoint conditioned on all previous datapoints. Given a trained model of this form, we are often interested in finding the model outputs that maximise the joint density i.e. we want $(x_1^*, \ldots, x_T^*) = \arg\max_{x_{1:T}} p(x_1, \ldots, x_T)$.

One simple method would be to use a greedy approach, sequentially choosing the output that maximises the probability at the given time step, conditioned on our choices at previous time steps:

$$\hat{x}_0 = \arg\max_{x_0} p(x_0) \tag{2.35}$$

$$\hat{x}_t = \arg\max_{x_t} p(x_t|\hat{x}_0, \ldots, \hat{x}_{t-1}) \qquad \text{for } t \in \{1, \ldots, T\} \tag{2.36}$$

While this approach is computationally efficient, the sequence $(\hat{x_0}, \ldots, \hat{x_T})$ is not guaranteed to be optimal.

Instead we could conduct an exhaustive search, explicitly calculating the probability for every output sequence. Although this is guaranteed to find the optimal sequence, this approach is often too computationally expensive. Suppose we want to find the optimal sentences of length $T = 20$ from a language model with a vocabulary of size 20,000. This would involve searching an output space of size $20,000^{20}$.

The beam search algorithm [Reddy, 1977] presents a powerful alternative. A parameter β, known as the beam width, is chosen such that at each time step only the top β outputs are stored, and all others ignored. On the first time step, we consider the β outputs with the highest probability. For the following time steps, we calculate the joint probability of the preceding stored sequences and the outputs at the current timestep, deleting all but the β most probable sequences.

Beam search provides a good compromise between the approaches above. It generally outputs sequences with a much higher probability density than those produced by the greedy approach, which corresponds to a beam search of width $\beta = 1$. Although the exhaustive approach will provide the optimal result, the computation cost of beam search is far lower.

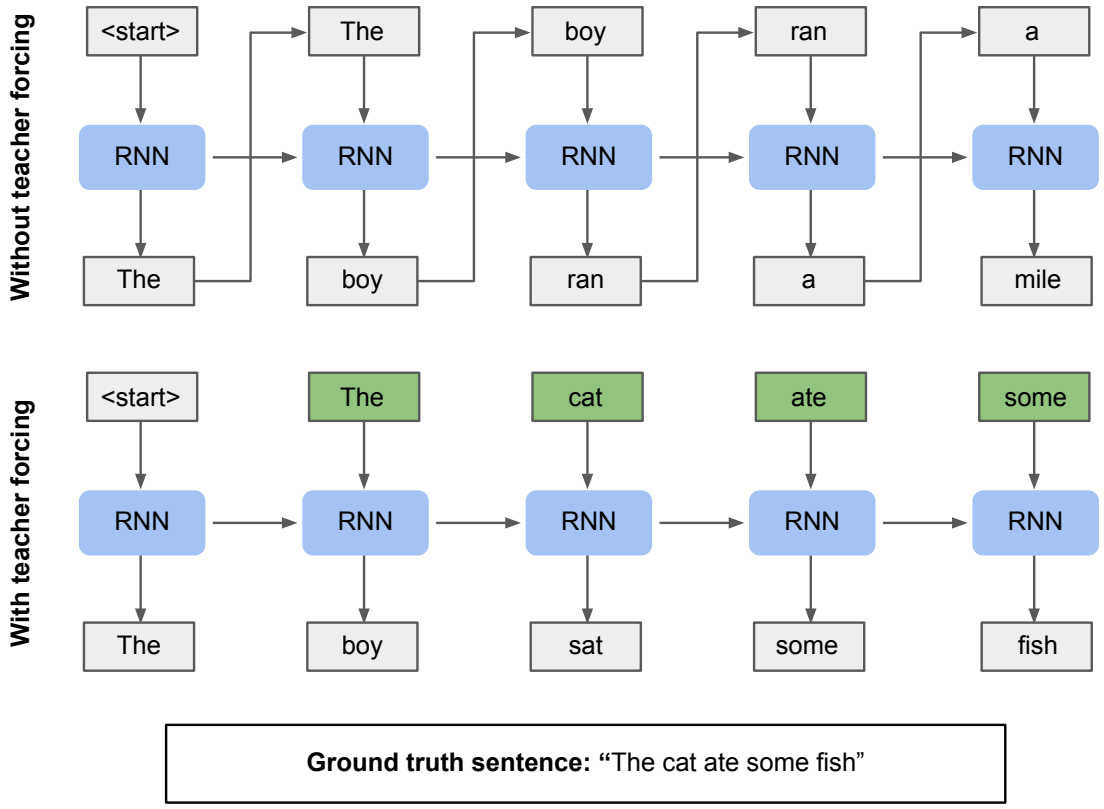


Figure 2.2: Schematic diagram comparing the RNN training without teacher forcing (top) and RNN training with teacher forcing (bottom).

2.2.2.2 Teacher Forcing

At test time, recurrent models condition the output at the current time step on model outputs from previous time steps as discussed in Section 2.2.2.1. The same approach can also be used during model training, feeding the model predictions from previous time steps back into the model before generating the next output, as shown in Figure 2.2 (top).

An alternative technique, known as teacher forcing [Williams and Zipser, 1989], feeds ground truth data points (rather than model predictions) from previous time steps into the model during training, as shown in Figure 2.2 (bottom). This prevents the RNN from deviating too far from the true sequence during training, which makes training significantly easier. We also note that if the model doesn't contain hidden to hidden connections, teacher forcing avoids the need to back propagate through time, although we consider only models that do contain such connections in this

work. While teacher forcing greatly improves training stability, some have reported an adverse effect when sampling from the model at test time, when model outputs (rather than ground truth data) must be fed back into the model. Further extensions have been proposed to mitigate this issue – see for example Goyal et al. [2016].

2.3 Differential Privacy

The primary goal of machine learning is to build models of observed data; training such models thus clearly necessitates access to such data. In many cases however, this data will be sensitive or confidential, and the data owner may be concerned about adversaries gaining sensitive information from their data. In a world where organisations are collecting personal data from billions of individuals every day, data privacy has never been more relevant.

The notion of data privacy refers to an individual's or organisation's ability to control the extent to which their data is used or observed by third parties. In many jurisdictions, data privacy is considered a fundamental human right, and violations can be considered criminal offences. Failure to protect data privacy can have countless negative repercussions. For example, it may enable fraudulent practices by adversaries, such as identity theft or phishing scams; it could expose sensitive or incriminating information about an individual (an example especially relevant to those living under repressive governments); or it could lead to unwanted algorithmic bias against an individual.

In order to protect data privacy, it is useful to first define a measure of privacy. A number of different measures have been proposed. A (concerningly) common approach is to 'anonymise' the data by removing identifiable information such as names or national identity numbers. However this approach can have catastrophic consequences, notably exposing dataset members to potential linkage attacks. These attacks aim to identify members by comparing data entries against members of other datasets. Famously, the medical records of the governor of Massachusetts were identified by comparing features from a publicly released (anonymised) insurance database with voting record data, as described in Sweeney [2002]. Similarly, the

identity of Netflix users were revealed by comparing members of the Netflix prize dataset with the Internet Movie Database (IMDb) [Narayanan and Shmatikov, 2007].

More rigorous, formal privacy definitions have been introduced offering stronger guarantees to individuals, including k-anonymity [Sweeney, 2002], ℓ-diversity [Machanavajjhala et al., 2006], differential privacy [Dwork et al., 2006], and metric privacy [Chatzikokolakis et al., 2013]. Throughout this work we focus on differential privacy, which is perhaps the most widely-adopted, and considered by many to be the gold standard of privacy guarantees.

Differential privacy defines a mathematically provable and quantifiable guarantee to the data owner, providing assurance that their sensitive information is not revealed to adversarial parties. Much of the research in differential privacy can be split into one of two models: the central model and the local model. Both provide powerful privacy guarantees to data owners, but each apply in different contexts. We outline these two models in the following sections.

2.3.1 Central Differential Privacy

The central model of differential privacy, commonly referred to as central differential privacy (CDP) is a framework for preventing adversaries from detecting the presence of an individual in a dataset. This guarantee is achieved via the careful addition of calibrated noise to the output of statistical queries on a dataset.

Definition 2.3.1. ((ϵ, δ)-central differential privacy) Let $\mathcal{M}^{(\text{central})} : \mathcal{D} \to \mathcal{Z}$ be a randomised algorithm, that takes as input datasets from the dataset domain $\mathcal{D}$. We say $\mathcal{M}^{(\text{central})}$ is (ϵ, δ)-central differentially private if for $\epsilon, \delta \geq 0$, for all subsets $S \subseteq \mathcal{Z}$, and for all neighbouring datasets $D, D' \in \mathcal{D}$, we have

$$p(\mathcal{M}^{(\text{central})}(D) \in S) \;\leq\; e^{\epsilon}\, p(\mathcal{M}^{(\text{central})}(D') \in S) + \delta \tag{2.37}$$

where for D and D' to be neighbouring means that they are identical in all but one datapoint. When $\delta = 0$, we say that $\mathcal{M}^{(\text{central})}$ is ϵ-central differentially private.

Intuitively, this states that one cannot tell (up to a level of certainty determined by ϵ and δ) whether an individual is present in a database or not, after observing the

output of such a query on the database. A more detailed discussion of this definition is given in Section 2.3.4.

Queries on datasets can take many forms – perhaps the simplest is a counting query, which outputs the number of members in a database that satisfy a given property. Consider a study attempting to find out how many people with a history of substance abuse suffer from long term health issues. Members of this substance abuse database may want to keep their presence in the database private to avoid, for example, potential adverse effects on their employment opportunities, or health insurance premiums, amongst other issues. CDP mechanisms can be used to privatise such queries.

Before introducing an appropriate CDP mechanism, we must first introduce the notion of sensitivity:

Definition 2.3.2. (ℓ_1 sensitivity): The ℓ_1 sensitivity of a function $f : \mathcal{D} \to \mathcal{Z}$ is defined as

$$\Delta f = \max_{\text{adjacent}(D,D')} ||f(D) - f(D')||_1 \tag{2.38}$$

where adjacent(D, D') implies $D, D' \in \mathcal{D}$ are neighbouring datasets.

Clearly, for a counting query, each database member contributes 1 to the output of the query if they satisfy the given property, or 0 otherwise. Thus if two datasets are identical in all but one entry, the overall count can differ by at most 1, i.e. $\Delta f = 1$. The Laplace mechanism can be used to privatise such a query:

Definition 2.3.3. (Laplace mechanism): The Laplace mechanism $\mathcal{M}^{(\text{central})} : \mathcal{D} \to \mathbb{R}^k$ is a randomised algorithm defined as

$$\mathcal{M}^{(\text{central})}(D, f(\cdot), \epsilon) = f(D) + (s_1, \ldots, s_k) \tag{2.39}$$

for $D \in \mathcal{D}$, $s_i \sim \text{Laplace}(0, \Delta f/\epsilon)$, and some transformation function $f : \mathcal{D} \to \mathbb{R}^k$.

Theorem 2.3.4. *The Laplace mechanism satisfies ϵ-central differential privacy.*

Proof. See Dwork and Roth [2014]. □

To privatise our counting query under ϵ-CDP, we must therefore add Laplace$(0, 1/\epsilon)$ noise to the final count.

In addition to providing a quantifiable measure of privacy, CDP also has some key properties. Notably, the post-processing theorem states that (data-independent) transformations to the output of a CDP mechanism cannot degrade the privacy guarantee. The composition theorem states how sequential querying of a dataset affects the privacy guarantees: specifically, the composition of an (ϵ_1, δ_1)-CDP query and an (ϵ_2, δ_2)-CDP query is at least $(\epsilon_1 + \epsilon_2, \delta_1 + \delta_2)$-CDP i.e. "the ϵ's and δ's add up". We also note that CDP provides protection against the aforementioned linkage attacks. We refer to Dwork and Roth [2014] for proofs of these results and a thorough overview of the central model.

While CDP has been in use for several decades, more recently this model has been applied to deep learning models. Most notably, with the introduction of CDP optimisation algorithms.

2.3.1.1 CDP Optimisation Algorithms

Neural networks have become the de facto standard for building powerful models of the underlying structure of data. Such models are generally trained with gradient-based optimisation methods. By construction, information about training set members will be 'leaked' into the model through gradient updates [Zhu et al., 2019], and ultimately contained within the parameters of the trained model. Model inversion attacks allow adversaries to access information about training set members from the trained model. Fredrikson et al. [2015] for example demonstrate how one can obtain recognisable images of training set member's faces given only their name and access to a facial recognition model.

The introduction of CDP optimisation algorithms such as Differentially Private Stochastic Gradient Descent (DP-SGD) [Abadi et al., 2016] and Differentially Private Adam (DP-Adam) [Gylberth et al., 2017], have allowed the training of deep neural networks under CDP. These techniques note that the calculation of gradients are a specific example of a query on the training dataset, and use a CDP mechanism to privatise each gradient update such that the final model satisfies CDP with respect

to the training set.

2.3.2 Local Differential Privacy

The local model of differential privacy, commonly referred to as local differential privacy (LDP), is related to the central model, but gives a much stricter guarantee. Rather than privatising queries on a (non-private) dataset, the local model is concerned with privatising individual datapoints. By allowing individuals to privatise their own data locally, they no longer need to rely on the trustworthiness of the database administrator. The trade-off is that, under this stricter guarantee, data utility is typically much lower than with approaches that use the central model (for the same value of ϵ).

To formalise the concept of LDP, we first introduce some definitions and notation.

Definition 2.3.5. (ϵ-local differential privacy) A local randomised algorithm $\mathcal{M} : \mathcal{X} \to \mathcal{Z}$, that takes as input a datapoint from the data domain $\mathcal{X}$, satisfies ϵ-local differential privacy if for $\epsilon \geq 0$, for any $S \subseteq \mathcal{Z}$, and for all inputs $x, x' \in \mathcal{X}$,

$$p(\mathcal{M}(x) \in S) \leq e^{\epsilon}\, p(\mathcal{M}(x') \in S) \tag{2.40}$$

This states that one cannot tell (with a level of certainty determined by ϵ) whether the output of a local randomised algorithm $\mathcal{M}$ is the privatised version of a datapoint $x \in \mathcal{X}$, or the privatised version of any other input $x' \in \mathcal{X}$.

In order to give a concrete example of how such a mechanism might work, we first introduce the notion of local sensitivity:

Definition 2.3.6. (Local ℓ_1 sensitivity) The local ℓ_1 sensitivity of a function $f : \mathcal{X} \to \mathcal{Z}$, where $\mathcal{Z} \subseteq \mathbb{R}^k$, is defined as

$$\Delta f = \max_{x,x' \in \mathcal{X}} ||f(x) - f(x')||_1 \tag{2.41}$$

Given this, we introduce perhaps the best-known LDP mechanism, the local Laplace mechanism:

Definition 2.3.7. (Local Laplace mechanism) The local Laplace mechanism $\mathcal{M} : \mathcal{X} \to \mathbb{R}^k$ is a randomised algorithm defined as

$$\mathcal{M}(x, f(\cdot), \epsilon) = f(x) + (s_1, \ldots, s_k) \tag{2.42}$$

for $x \in \mathcal{X}$, $s_i \sim \text{Laplace}(0, \Delta f/\epsilon)$, and some transformation function $f : \mathcal{X} \to \mathcal{Z}$ with local ℓ_1 sensitivity Δf.

Theorem 2.3.8. *The local Laplace mechanism satisfies ϵ-local differential privacy.*

Proof. We follow an approach similar to the proof in Dwork and Roth [2014] that the central Laplace Mechanism satisfies CDP. Assume $x \in \mathcal{X}$ and $x' \in \mathcal{X}$ are two arbitrary datapoints. Denote $\mathcal{M}(x) = f(x) + (s_1, \ldots, s_k)$ where $s_i \sim \text{Laplace}(0, \Delta f/\epsilon)$. Then for some arbitrary c we know that

$$\frac{p(\mathcal{M}(x) = c)}{p(\mathcal{M}(x') = c)} = \prod_{i=1}^{k} \frac{p(\mathcal{M}_i(x) = c_i)}{p(\mathcal{M}_i(x') = c_i)} \tag{2.43}$$

$$= \prod_{i=1}^{k} \frac{e^{-\frac{\epsilon|f_i(x)-c_i|}{\Delta f}}}{e^{-\frac{\epsilon|f_i(x')-c_i|}{\Delta f}}} \tag{2.44}$$

$$= \prod_{i=1}^{k} e^{\frac{\epsilon(|f_i(x')-c_i|-|f_i(x)-c_i|)}{\Delta f}} \tag{2.45}$$

$$\leq \prod_{i=1}^{k} e^{\frac{\epsilon|f_i(x')-f_i(x)|}{\Delta f}} \tag{2.46}$$

$$= e^{\frac{\epsilon||f(x')-f(x)||_1}{\Delta f}} \tag{2.47}$$

$$\leq e^{\epsilon} \tag{2.48}$$

where the first inequality comes from the triangle inequality, and the second comes from the definition of Δf. □

2.3.3 Central vs. Local Differential Privacy

While both central and local differential privacy provide powerful privacy guarantees to individuals, it is important to distinguish between the two. The key distinction is that the central model assumes a database administrator has access to unprivatised,

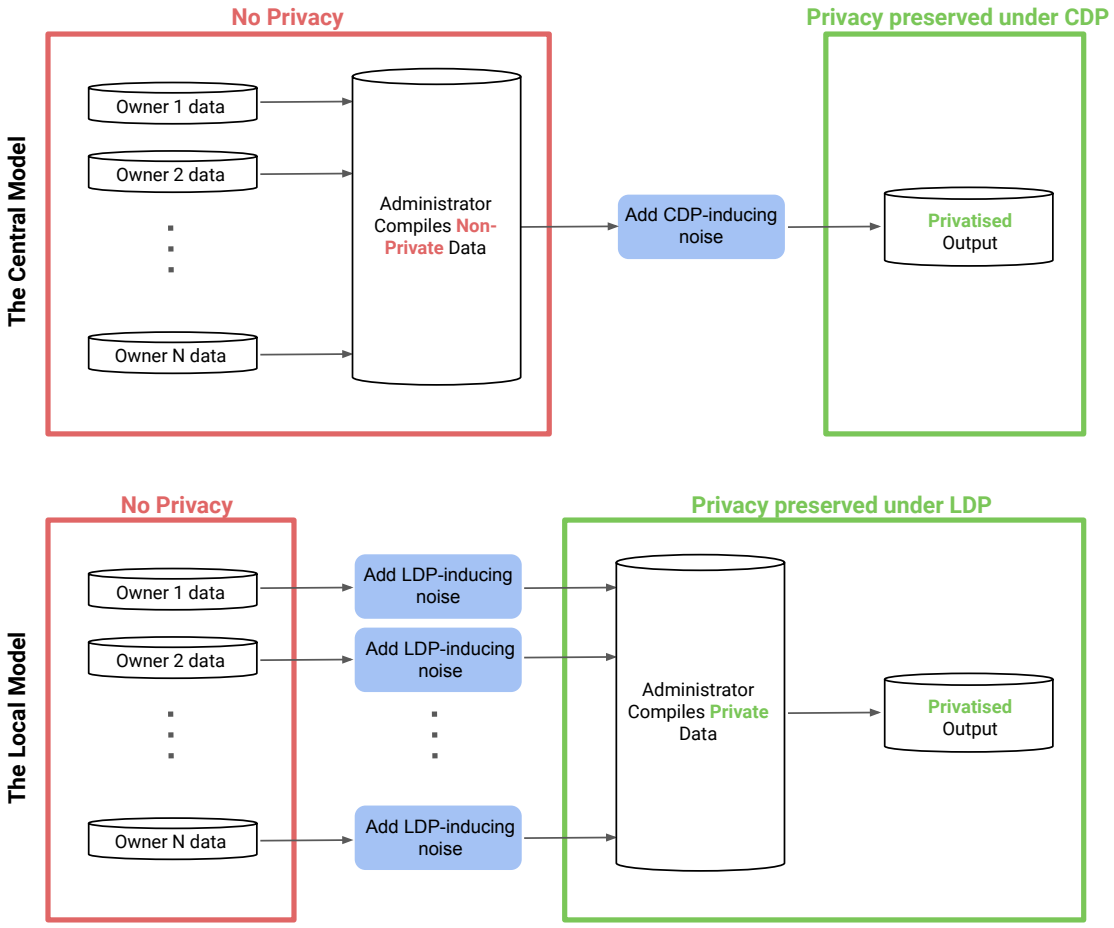


Figure 2.3: Schematic diagram comparing the central model of differential privacy (top) with the local model (bottom).

sensitive data. The administrator applies the privatisation mechanism to transformations of this sensitive data. The central model is therefore only applicable when the data owner considers the administrator both trustworthy, and sufficiently protected from failures such as hacking.

The local model does not make any such assumptions. Rather, this model puts total control in the hands of individual data owners. Each data owner privatises their data locally *before* sharing with the database administrator. Since nobody but the individual has access to their own sensitive features, this model is applicable in the absence of trustworthy third parties.

It is clear that, in many real-life scenarios where companies and organisations are collecting sensitive user data, the local model provides a much greater level of protection to the individual than the local model. Figure 2.3 shows a schematic

diagram of the privatisation procedure for the central model (top) and the local model (bottom).

2.3.4 Interpretation of ϵ

It is clear from Definitions 2.3.1 and 2.3.5 that ϵ controls the trade-off between data privacy and data utility: lower ϵ values correspond to stronger guarantees but suffer from poor data utility; higher ϵ values allow improved utility but at the expense of weaker privacy guarantees.

Noting the symmetry in the definition of CDP (with $\delta = 0$), Equation 2.37 can be re-written as

$$e^{-\epsilon}\, p(\mathcal{M}^{(\text{central})}(D') \in S) \;\leq\; p(\mathcal{M}^{(\text{central})}(D) \in S) \tag{2.49}$$

and thus combining Equations 2.37 and 2.49, we can write

$$|\log p(\mathcal{M}^{(\text{central})}(D) \in S) - \log p(\mathcal{M}^{(\text{central})}(D') \in S)| \leq \epsilon \tag{2.50}$$

which gives us a somewhat more interpretable meaning of ϵ. The presence of any given individual in the database can change the log probability of observing any query output $S \subseteq \mathcal{Z}$ by at most ϵ.

Similarly for LDP, Equation 2.40 can be re-written as

$$|\log p(\mathcal{M}(x) \in S) - \log p(\mathcal{M}(x') \in S))| \leq \epsilon \tag{2.51}$$

implying that the distributions over outputs $S \subseteq \mathcal{Z}$ are similar (indeed the log probabilities do not differ by more than ϵ) regardless of which true underlying datapoint was passed into the mechanism.

Determining a suitable value of ϵ has mostly been discussed in the context of CDP: Lee and Clifton [2011] note how the guarantee enforced by a given level of ϵ will vary between data domain and query type. They discuss the guarantee imposed by ϵ in terms of its relation to the privacy risk (i.e. the probability of an individual being identified as present or absent in a dataset) – a model further generalised by

Mehner et al. [2021]. Alternative approaches have been proposed by Hsu et al. [2014] and Krehbiel [2019], although in general, a method of determining a suitable value of ϵ remains an open question in the literature.

Table 1 in Hsu et al. [2014] gives an overview of ϵ values used by CDP mechanisms in the literature, ranging from 0.01 to 10, though some work considers values up to 100 [Yu et al., 2014]. A rule of thumb widely used in the CDP literature is to use 'single digit ϵ' (i.e. $\epsilon \leq 10$).

LDP provides much stricter guarantees than CDP, and higher ϵ values are generally accepted in practical applications, due to the otherwise poor data utility in low ϵ settings. Little work has been done on suitable values of ϵ specifically for LDP, though single digit ϵ values are again considered the de facto standard in the literature [Duchi et al., 2018, Wang et al., 2019]. Bhowmick et al. [2019] assume that the adversary does not have access to all data but are 'curious onlookers' who have little prior information on individuals; they argue that at least in their federated learning setting, much higher values of ϵ are then acceptable, and experiments consider $\epsilon \geq 50$. In this thesis, we introduce several LDP mechanisms and study their performance at a broad range of ϵ values, whilst adhering to the commonly used single digit ϵ rule of thumb (i.e. $\epsilon \leq 10$).

Chapter 3

Powerful Latent Representations for High-Dimensional Data

The work presented in this chapter was published in [Mansbridge et al., 2019].

In the previous chapter, we gave a concise overview of the background material required for the remaining chapters in this thesis.

In this chapter, we introduce a novel latent variable model designed to encourage richer representations in the latent space. Powerful generative models, particularly in natural language modelling, are commonly trained by maximising a variational lower bound on the data log likelihood. These models often suffer from poor use of their latent variable, with ad-hoc annealing factors used to encourage retention of information in the latent variable. Instead, we propose encoding powerful latent representations through an objective that encourages both generation and reconstruction of the input data – this objective is reliant on information being contained within the latent.

3.1 Introduction

Language modelling has remained a challenging problem in the research community for several decades: learning from unlabelled data is a difficult problem, and language data is inherently sparse and high dimensional. Latent variable models, particularly in the form of Variational Autoencoders (VAEs) [Kingma and Welling, 2014, Rezende

et al., 2014], have been employed in natural language modelling tasks using varied architectures for both the encoder and the decoder [Bowman et al., 2016, Dieng et al., 2017, Semeniuta et al., 2017, Yang et al., 2017, Shah et al., 2018]. However, a model utilising powerful architectures that is able to effectively capture meaningful semantic information in its latent variables is yet to be discovered.

A VAE approach to language modelling was given by Bowman et al. [2016], the graphical model for which is shown in Figure 3.1a. This forms a generative model of a sentence x, conditioned on a latent variable z. As discussed in Section 2.1.1.2, the integral $p_\theta(x) = \int p_\theta(x|z)p(z)dz$ is typically intractable, and so the Evidence Lower Bound (ELBO) on the log likelihood is maximised:

$$\log p(x) \geq \mathbb{E}_{q_\phi(z|x)}\big[\log p_\theta(x|z)\big] - D_{\mathrm{KL}}\big[q_\phi(z|x)||p(z)\big] \tag{3.1}$$

where summing over all datapoints x gives a lower bound on the likelihood of the full dataset.

3.2 Posterior Collapse

In language modelling, typically both the generative model $p_\theta(x|z)$ and approximate posterior distribution $q_\phi(z|x)$, are parameterised using an LSTM recurrent neural network – see for example Bowman et al. [2016]. This autoregressive generative model is so powerful that the optimum of Equation 3.1 is typically achieved without making appreciable use of the latent variable in the model – a phenomenon referred to as posterior collapse.

Examining Equation 3.1 it becomes clear why this occurs. Maximising the lower bound is equivalent to jointly maximising the reconstruction term $\mathbb{E}_{q_\phi(z|x)}\big[\log p_\theta(x|z)\big]$, and minimising the KL divergence term $D_{\mathrm{KL}}\big[q_\phi(z|x)||p(z)\big]$. By definition, the KL divergence is lower bounded by zero, and this zero value occurs when the approximate posterior $q_\phi(z|x)$ learns to match the prior $p(z)$. If a small generative network is used, the generative model will be reliant on information encoded in the latent in order to maximise the reconstruction term. Therefore, $q_\phi(z|x)$ is discouraged from

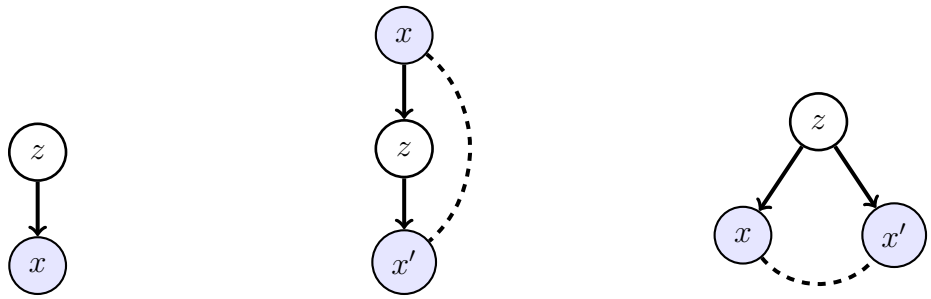
matching the prior and the KL term will be non-zero. However, powerful generative models are able to sample meaningful datapoints (maximising the reconstruction term) without use of the latent. In this scenario, the model would typically converge to an optimum where $q_\phi(z|x)$ 'collapses' to the prior, minimising the KL divergence. When this happens, a fully observed model has essentially been learnt, and any advantages that motivated the use of a latent variable model have been lost.

The dependency between what is represented by latent variables, and the capacity of the decoding distribution (i.e., its ability to model the data without using the latent) has been studied in the literature. For example, Yang et al. [2017] use a lower capacity dilated convolutional decoder to generate sentences in an effort to avoid posterior collapse, and Gulrajani et al. [2017] discuss these effects in the context of image processing.

In the experiments in Section 3.4.2 we will see evidence of this phenomenon in practice: we demonstrate that the VAE is incapable of reconstructing data since the latent variable, which forms a bottleneck, contains little information on the input data fed into the approximate posterior distribution.

Most commonly, a training procedure called 'KL annealing' is used to avoid posterior collapse, in which the KL divergence term in the objective is slowly turned on during training [Bowman et al., 2016, Sønderby et al., 2016]. KL annealing allows the model to use its latent variable to some degree by forcing the model into a local maximum of its objective function. An alternative approach to encourage use of the latent is to aggressively optimise the inference network, performing multiple updates of the posterior network parameters ϕ between every update of the generative parameters θ [He et al., 2019].

If one must reduce model capacity or modify the training procedure in this way to preferentially obtain local maxima, this suggests that the objective function in Equation 3.1 may not be well-suited to learning latent representations of high-dimensional data.



(a) (b) (c)

Figure 3.1: (a) Standard generative model. (b) Stochastic autoencoder with tied observations. (c) Equivalent tied stochastic autoencoder with AutoGen parameterisation.

3.3 High-Fidelity Latent Variable Modelling with AutoGen

We propose a new generative latent variable model, motivated by the autoencoder framework [Hinton and Zemel, 1994, Hinton and Salakhutdinov, 2006], designed to encourage powerful latent encodings. Autoencoders are trained to reconstruct data through a low-dimensional bottleneck layer, and as a result, construct a dimensionally-reduced representation from which the data can be reconstructed. By encouraging reconstructions in our model, we force the latent variable to represent the input data, overcoming the issues faced by VAEs [Bowman et al., 2016] where the latent variable is ignored, as discussed in Section 3.1.

To autoencode in a probabilistic model, we start by considering a 'stochastic autoencoder' (SAE) in which we maximise the likelihood of a reconstruction:

$$p_{\text{SAE}}(x' = x_n | x = x_n) = \int p_{\text{SAE}}(x' = x_n | z)\, p_{\text{SAE}}(z | x = x_n)\, dz \tag{3.2}$$

where x' represents the reconstruction and the training data is denoted by $\{x_n\}$. Maximising this likelihood would encourage high-fidelity reconstruction from the stochastic embedding z by tying the input data x and the output x', much like an autoencoder. The associated graphical model is shown in Figure 3.1b.

However, it is not immediately clear how to train such a model – constructing

a lower bound on the likelihood using variational methods common in the VAE literature will give rise to an intractable $p(x)$ term. This SAE would also not allow generation from a prior distribution, as in the case of VAEs. In order to leverage both prior generation and high-fidelity reconstruction from the latent variable, we propose jointly maximisng the likelihood of a SAE and a VAE under a set of assumptions that tie the two models together:

$$\mathcal{L}_{\text{AutoGen}} = \sum_n \underbrace{\log p_{\text{VAE}}(x = x_n)}_{\text{generation}} + \underbrace{\log p_{\text{SAE}}(x' = x_n | x = x_n)}_{\text{reconstruction}} \tag{3.3}$$

The reconstruction term is given in Equation 3.2, and we can write the generation term as

$$p_{\text{VAE}}(x = x_n) = \int p_{\text{VAE}}(x = x_n | z)\, p_{\text{VAE}}(z)\, dz \tag{3.4}$$

Crucially, maximising $\mathcal{L}_{\text{AutoGen}}$ does not correspond to maximising the log likelihood of the data as in the case of a VAE, nor would a lower bound on $\mathcal{L}_{\text{AutoGen}}$ correspond to the VAE ELBO (Equation 3.1). Instead, we will see that $\mathcal{L}_{\text{AutoGen}}$ represents the log likelihood of a different model that combines both VAEs and SAEs.

As yet, we have not specified the relationship between the two terms in $\mathcal{L}_{\text{AutoGen}}$, given by Equations 3.2 and 3.4. First, we assume that the generative model $p_{\text{VAE}}(x = x_n|z)$ in the VAE is the same as the reconstruction model $p_{\text{SAE}}(x' = x_n|z)$ in the SAE, and that the two models share a prior: $p_{\text{SAE}}(z) = p_{\text{VAE}}(z)$. Under this equality assumption, it makes sense to denote these distributions identically as $p(x = x_n|z)$ and $p(z)$, respectively. Second, we assume that the encoding and decoding distributions in the stochastic autoencoder are symmetric. Using Bayes' rule, we write these assumptions as

$$p_{\text{SAE}}(z|x = x_n) \overset{\text{sym. assump.}}{=} \frac{p_{\text{SAE}}(x' = x_n|z)\, p_{\text{SAE}}(z)}{p_{\text{SAE}}(x = x_n)} \tag{3.5}$$

$$\overset{\text{eq. assump.}}{=} \frac{p(x = x_n|z)\, p(z)}{p(x = x_n)} \tag{3.6}$$

These assumptions constrain the two otherwise-independent models, allowing AutoGen to demand both generation from the prior (like VAEs) and high-fidelity

reconstructions from the latent (like autoencoders), all while specifying a single generative model, $p(x = x_n|z)$.

Indeed, using this equality assumption allows us to write $p_{\text{SAE}}(x = x_n) = p_{\text{VAE}}(x = x_n) =: p(x = x_n)$. Thus, we can write Equation 3.3 as:

$$\mathcal{L}_{\text{AutoGen}} = \sum_n \left[\log p(x = x_n) + \log \int dz\, p(x = x_n|z) p_{\text{SAE}}(z|x = x_n) \right] \tag{3.7}$$

Now applying Equation 3.6 and combining the two logarithms, we find

$$\mathcal{L}_{\text{AutoGen}} = \sum_n \log \int dz\, p(x = x_n|z)^2 p(z) \tag{3.8}$$

In other words, AutoGen can be interpreted as the tying of two separate generations from the same model $p(x = x_n|z)$. The graphical representation of this interpretation is shown in Figure 3.1c, where the dashed line corresponds to the tying (equality) of the two generations.

With the AutoGen assumptions, a simple lower bound for $\mathcal{L}_{\text{AutoGen}}$ can be derived from Equation 3.8, following the standard variational lower bound arguments:

$$\mathcal{L}_{\text{AutoGen}} \geq \sum_n 2\, \mathbb{E}_{q(z|x_n)} \big[\log p(x = x_n|z) \big] - D_{\text{KL}} \big[q(z|x_n) || p(z) \big] \tag{3.9}$$

3.3.1 Multiple Reconstructions

We see that the variational lower bound derived for AutoGen in Equation 3.9 is the same as that of the VAE [Kingma and Welling, 2014, Rezende et al., 2014], but with a factor of 2 in front of the reconstruction term. It is important to emphasise, however, that the AutoGen objective is not a lower bound on the data log likelihood. Equation 3.9 is a lower bound on the sum of the log likelihoods in Equation 3.3, and represents a criterion for training a generative model $p(x|z)$ that evenly balances both good spontaneous generation of the data $p(x = x_n)$ as well as high-fidelity reconstruction $p(x' = x_n|x = x_n)$.

Of course, AutoGen does not force the latent variable to encode information in a particular way, but it is a necessary condition that the latent represents the data

well in order to reconstruct it. We discuss the relation between AutoGen and other efforts to influence the latent representation of VAEs in Section 3.5.

A natural generalisation of the AutoGen objective and assumptions is to maximise the log likelihood of m independent-but-tied reconstructions, instead of just one. The arguments above then lead to a lower bound with a factor of $1 + m$ in front of the generative term:

$$\mathcal{L}_{\text{AutoGen}}(m) \;\geq\; \sum_{n} (1+m)\, \mathbb{E}_{q(z|x_n)}\big[\log p(x_n|z)\big] - D_{\text{KL}}\big[q(z|x_n)||p(z)\big] \tag{3.10}$$

Larger m encourages better reconstructions at the expense of poorer generation. We discuss the impact of the choice of m in Section 3.4.

3.4 Experiments

We train four separate language models, each with the posterior and generative distributions parameterised by LSTM networks as in Bowman et al. [2016]. Two of these models are VAEs – one such variant uses KL annealing, and the other does not. We then train our baseline AutoGen model, which uses the objective in Equation 3.9, and train an AutoGen variant using the objective in Equation 3.10 with $m = 2$.

All of the models were trained using the BookCorpus dataset [Zhu et al., 2015], which contains sentences from a collection of 11,038 books. We restrict our data to contain only sentences with length between 5 and 30 words, and restrict our vocabulary to the most common 20,000 words. We use 90% of the data for training and 10% for testing. After pre-processing, this equates to 58.8 million training sentences and 6.5 million test sentences. All models in this section are trained using word drop as in Bowman et al. [2016].

Neither AutoGen models are trained using KL annealing. We consider KL annealing to be unprincipled, as it destroys the relevant lower bound during training. In contrast, AutoGen provides an unfettered lower bound throughout training. Despite not using KL annealing, we show that AutoGen improves latent-variable descriptiveness compared to VAEs both with and without KL annealing for completeness.

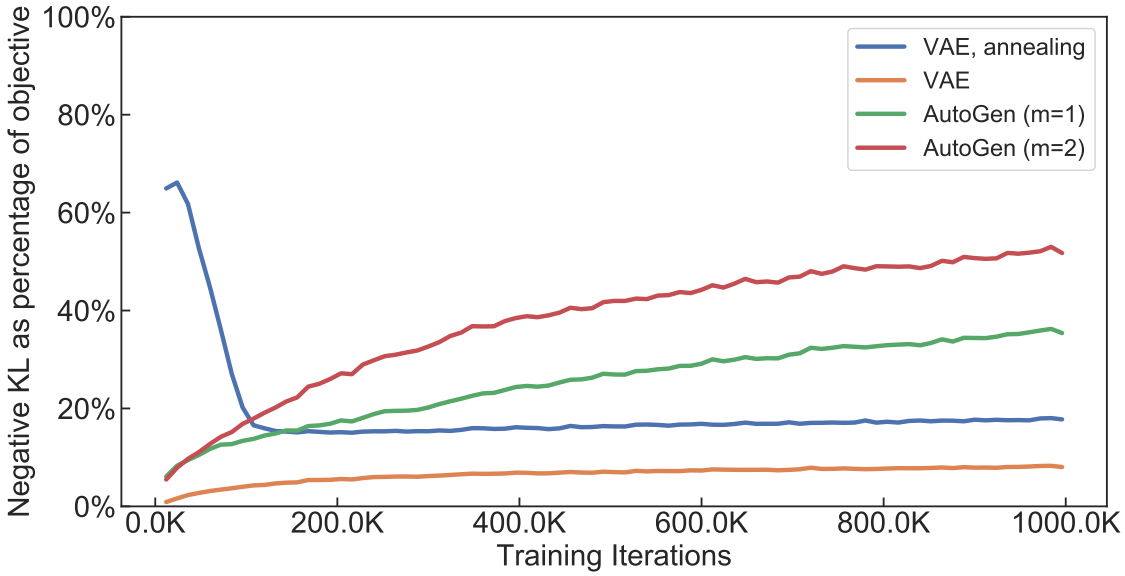


Figure 3.2: Negative $D_{\mathrm{KL}}\big[q(z|x_n)||p(z)\big]$ term as a % of overall objective for the four models throughout training.

3.4.1 Optimisation Results

We train all models for 1 million iterations using mini-batches of 200 sentences. We use 500 hidden states for the LSTM cells in our encoder and decoder networks, and dimension 50 for our latent variable z. The objective functions differ between the four models, and so it is not meaningful to directly compare them. Instead, in Figure 3.2, we show the % of the objective function that is accounted for by the (negative) KL term. Despite the fact that AutoGen has a larger pre-factor in front of the reconstruction term, the KL term becomes more and more significant with respect to the overall objective function for AutoGen with $m = 1$ and $m = 2$, as compared to the VAE. This suggests that the latent in AutoGen is putting less emphasis on matching the prior $p(z)$, emphasising instead the representation of the data.

To understand the impact of AutoGen on the log likelihood of the training data (the generation term in Equation 3.3), we compare the VAE ELBO in Equation 3.1 of the four models during training. Since the ELBO is the objective function for the VAE, we expect it to be a relatively tight lower bound on the log likelihood. However, this only applies to the VAE. Indeed, if the VAE ELBO calculated with the AutoGen model is similar to that of the VAE, we can conclude that the AutoGen

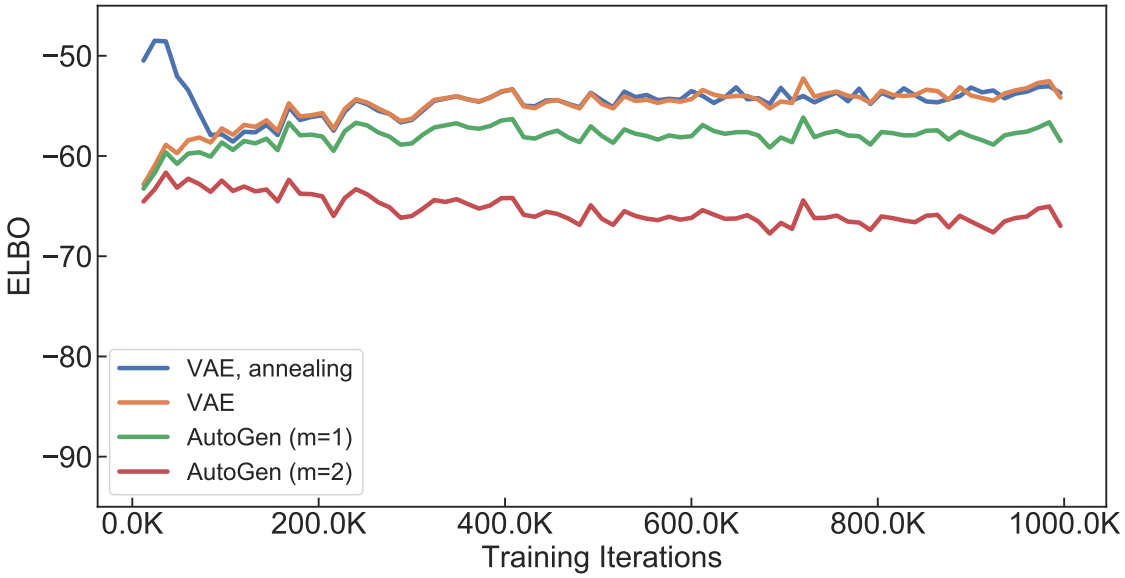


Figure 3.3: ELBO (log likelihood lower bound, Equation 3.1) for the four models throughout training.

model is approximately concurrently maximising the log likelihood as well as its reconstruction-specific objective.

In Figure 3.3 we show the ELBO for all four models. We see that, though the baseline AutoGen ($m = 1$) ELBO is below that of the VAE, it tracks the VAE ELBO well and is non-decreasing. On the other hand, for the more aggressive AutoGen with $m = 2$, the ELBO starts decreasing early on in training and continues to do so as its objective function is maximised. Thus, for the baseline AutoGen with objective function corresponding to maximising Equation 3.3, we expect decent reconstructions without significantly compromising generation from the prior. However, we will likely see some degradation in the ability of AutoGen ($m = 2$) to generate meaningful samples. In Sections 3.4.2 and 3.4.3 we corroborate this expectation qualitatively by studying samples from the models.

3.4.2 Sentence Reconstruction

Indications that AutoGen should more powerfully encode information into its latent variable were given theoretically in the construction of AutoGen in Section 3.3, and from the optimisation results in Section 3.4.1. To see what this means for explicit samples, we perform a study of the sentences reconstructed by the VAE compared

Table 3.1: Reconstructed sentences from the VAE (top) and AutoGen (bottom). Sentences are not 'cherry picked': these are the first four sentences reconstructed from a grammatically correct input sentence, between 4 and 20 words in length (for aesthetics), and with none of the sentences containing an unknown token (for readability). All punctuation is generated by the models.

Input Sentence	VAE Reconstruction	VAE Reconstruction (annealing)
"more or less?"	"oh yeah." "	"what about you?"
why would you need to talk when they can do it for you?	how couldn't i?	why do you want to know if i can find out of here?
she had no idea how beautiful she truly was.	she hadn't.	she had no idea what she was talking about.
"i guess some people never learn."	"i love you.	"you know what you're thinking."

Input Sentence	AutoGen Reconstruction ($m = 1$)	AutoGen Reconstruction ($m = 2$)
"more or less?"	"more or less?"	"more or less?"
why would you need to talk when they can do it for you?	why would you need to know if you can do it for you?	why would you need to talk when they can do it for you?
she had no idea how beautiful she truly was.	she had no idea how beautiful she was to.	she had no idea how beautiful she truly was.
"i guess some people never learn."	"i guess our parents never exist.	"i guess some people never learn."

to those reconstructed by AutoGen.

Table 3.1 shows reconstructions of input sentences x from the test set; each reconstruction is obtained via the maximisation of $p(x|z)$, as determined using beam search with beam width 15 (for details, see Section 2.2.2.1). We sample $z \sim q(z|x)$ in this process, meaning we find different reconstructions every time from the same input sentence, despite the beam search procedure in the reconstruction.

AutoGen is qualitatively better at reconstructing sentences than the VAE. For AutoGen with $m = 2$ we see that all sentences are reconstructed verbatim. For AutoGen with $m = 1$, even when the input sentence is not reconstructed verbatim, the model is able to generate a coherent sentence with a similar meaning by using semantically similar words. For example in the last sentence, by replacing "some people" with "our parents", and "never learn" with "never exist". On the other hand, the VAE reconstructions regularly produce sentences that have little relation to the input. Note that without annealing, the VAE regularly ignores the latent,

Table 3.2: Results from a blind survey comparing reconstruction quality. Respondents were told to "choose the best reconstruction", and where ambiguous, could discard sentence pairs.

Model 1 vs. Model 2	% responses with Model 1 as winner
VAE (annealing) vs. VAE	66%
AutoGen ($m = 1$) vs. VAE (annealing)	88%
AutoGen ($m = 2$) vs. AutoGen ($m = 1$)	88%

producing short, high-probability sentences reconstructed from the prior.

To make these results more quantitative, we ran three versions of a survey in which respondents were asked to judge the best reconstructions from two models. In the first survey, we received responses from 6 people who compared 120 pairs of reconstructions from the VAE and the VAE with annealing. The second survey received responses from 13 people over 260 sentences and compared reconstructions from the VAE with annealing to AutoGen ($m = 1$). The third compared AutoGen ($m = 1$) to AutoGen ($m = 2$) and received 23 responses over 575 sentences. None of the respondents in these surveys were authors of this work. The surveys were designed in this way to provide an easy binary question for the respondents. They provide a suitable test of the models due to the transitive nature of the comparisons.

Our survey results are shown in Table 3.2. We can clearly see that AutoGen with $m = 2$ outperforms AutoGen with $m = 1$, as expected. Similarly, AutoGen with $m = 1$ outperforms the VAE with annealing, and this in turn outperforms the standard VAE. All results have greater than 99% confidence.

3.4.3 Sentence Generation

The objective function of AutoGen encourages the generation of higher-fidelity reconstructions from its approximate posterior. The fundamental trade-off is that it may be less capable of generating sentences from its prior.

To investigate the qualitative impact of this trade-off, we now generate samples from the prior $z \sim \mathcal{N}(0, \mathrm{I})$ of the VAE and AutoGen. For a given latent z, we obtain sentences via beam search as in Section 3.4.2. Results are shown in Table 3.3, where

Table 3.3: Sentences generated from the prior, $z \sim \mathcal{N}(0, \mathrm{I})$, for the VAE (top) and AutoGen (bottom). Sentences are not 'cherry picked': they are chosen in the same way as those in Table 3.1. All punctuation is generated by the models.

VAE Generation	VAE Generation (annealing)
the only thing that mattered.	she just looked up.
he gave her go.	she felt her lips together.
"good morning," i thought.	my hands began to fill the void of what was happening to me.
she turned to herself.	at first i knew he would have to.

AutoGen Generation ($m = 1$)	AutoGen Generation ($m = 2$)
they don't show themselves in mind, or something to hide.	jack was moving with slow, jerky as strained to keep pace.
her eyes widen, frowning.	"i know that over something i don't answer."
the lights lit up around me.	the car jerked and splashed out the fuel fire hole.
i just feel like fun.	i turned disbelieving.

we see that there appears to be no obvious qualitative difference between the VAE with annealing and AutoGen at $m = 1$, with both models able to generate similarly coherent sentences. We note however that the generation quality seems to deteriorate for AutoGen at $m = 2$.

To be more quantitative, we ran a survey of 23 people – none of which were the authors – considering 392 sentences generated from the priors of all four of the models under consideration. We applied the same sentence filters to these generated sentences as we did to those generated in Table 3.3. We then asked the respondents whether or not a given sentence "made sense", maintaining the binary nature of the question, but allowing the respondent to interpret the meaning of a sentence "making sense". To minimise systematic effects, each respondent saw a maximum of 20 questions, evenly distributed between the four models. All sentences in the surveys were randomly shuffled with the model information obfuscated.

The results of our survey are shown in Table 3.4. Since the VAE generates systematically shorter sentences than the training data, which are inherently more likely to be meaningful, we divide our survey into short and long sentences (with length ≤ 10 and > 10 tokens, respectively). We conclude that the VAE with annealing is better at generating short sentences than AutoGen ($m = 1$). However, both models

Table 3.4: Results from a blind survey testing generation quality. Respondents were asked "does this sentence make sense" for a randomised list of sentences evenly sampled from the four models. Results are split into two sentence lengths L in order to mitigate the bias of the VAE models to generate short sentences.

Model	% meaningful ($L \leq 10$)	% meaningful ($L > 10$)
VAE	75%	N/A
VAE (annealing)	76%	32%
AutoGen ($m = 1$)	50%	32%
AutoGen ($m = 2$)	29%	5%

achieve equal results on generation quality for longer sentences. AutoGen is likely able to effectively generate longer sentences as a result of its more informative latent representation, which reduces the burden on the generative model of modelling long-term dependencies.

We see that AutoGen ($m = 2$) generates less meaningful sentences than other models. This is as expected since the training objective places less emphasis on generation quality. Finally, we emphasise that the baseline VAE could not generate *any* long sentences, bringing into question its merits as a generative language model. All results that differ by more than 1 percentage point in the table are statistically significant with confidence greater than 99%.

3.4.4 Latent Manifold Structure

Finally, with high-fidelity reconstructions from the latent, one would expect to be able to witness the smoothness of the latent space well. This seems to be the case, as can be seen in Table 3.5, where we show the reconstructions of a linear interpolation between two encoded sentences for VAE with annealing and for AutoGen ($m = 1$). The AutoGen interpolation seems to be qualitatively smoother: while neighbouring sentences are more similar, there are fewer instances of reconstructing the same sentences at subsequent interpolation steps.

The reconstructions from the VAE without annealing have little dependence on the latent, and AutoGen ($m = 2$) struggles to generate from the prior. As a consequence, both of these models show highly non-smooth interpolations with little similarity between subsequent sentences. The results for these models have therefore

Table 3.5: Latent variable interpolation. Two sentences (first and last sentences shown) are randomly selected from the test dataset and encoded into z_1 and z_2. Sentences are then generated along 10 evenly spaced steps from z_1 to z_2. This interpolation was not 'cherry picked': it was our first generated interpolation using the same filters as in previous tables. All punctuation is generated by the models.

VAE (annealing)	AutoGen ($m = 1$)
"i'll do anything, blake."	"i'll do anything, blake."
"i'll be right back then."	"i'll do it, though."
"i'll tell me like that."	"i'll say it, sir."
i dont know what to say.	"i've done it once."
i dont know what to say.	i dont think that was it.
i dont think about that way.	i wish so, though.
i'll be right now.	i bet it's okay.
i was so much.	i know how dad.
i looked at him.	i laughed at jack.
i looked at him.	i looked at sam.
i looked at adam.	i looked at adam.

been omitted.

We have provided only a single sample interpolation, and though it was not cherry picked, we do not attempt to make a statistically significant statement on the smoothness of the latent space. Given the theoretical construction of AutoGen, and the robust results shown in previous sections, we consider smoothness to be expected. The sample shown is consistent with our expectations, though we do not consider it a definite empirical result.

3.5 Discussion

We have introduced AutoGen, a novel latent variable model trained through the optimisation of an objective function specifically designed to encode information into the latent variable. AutoGen, by design, overcomes a widely observed shortcoming of VAEs whereby they can learn to generate data without encoding any information into the latent. AutoGen does so in a principled way, by explicitly modelling both the generation, and high-fidelity reconstruction, of the data. This is especially useful when the generative model is powerful – for example, when parameterised by an

autoregressive LSTM.

Other work towards enabling a VAE's latent variable to learn meaningful representations has focused on managing the structure of the representation, such as ensuring disentanglement. A detailed discussion of disentanglement in the context of VAEs is given by Higgins et al. [2017] and its references. Gulrajani et al. [2017] present an approach for controlling what is encoded into the latent in the context of image generation – the authors restrict the decoding network such that it models only local information in the image (e.g., texture, shading), allowing their latents to describe global information (e.g., object geometry, overall colour).

Demanding high-fidelity reconstructions from latent variables in a model, as in our approach, is in tension with demanding specific information to be stored in the latent variables (e.g. disentanglement). This can be seen very clearly by comparing our work to Higgins et al. [2017], where the authors introduce an ad-hoc factor of β in front of the KL divergence term in the VAE objective function. They find that $\beta > 1$ is required to improve the disentanglement of their latent representations.

Interestingly, $\beta > 1$ corresponds analytically to $-1 < m < 0$ in Equation 3.10. Equivalently, since the overall normalisation of the objective function does not impact the location of its extrema, Equation 3.10 is equivalent to the β-VAE objective function with $\beta = (1 + m)^{-1}$.

Since m in AutoGen represents the number of times a high-fidelity reconstruction is demanded (in addition to a single generation from the prior), β-VAE with $\beta > 1$ is analytically equivalent to demanding a '*negative* number of reconstructions'. As an analytic function of m, with larger m corresponding to higher-fidelity reconstructions, negative m would correspond to a deprecation of the reconstruction quality. This is indeed what the authors in Higgins et al. [2017] find and discuss. They view β-VAE as a technique to trade off more disentangled representations at the cost of lower-fidelity reconstructions, in contrast to our view of AutoGen as a technique to trade off higher-fidelity reconstructions at the cost of slightly inferior generation from the prior.

In connecting to β-VAE, we have considered AutoGen with m as a real number. Practically, m could take positive real values, and can be seen as a hyperparameter

that requires task-specific tuning. From our results, we expect $m \approx 1$ to be a useful ballpark value, with smaller m improving generation from the prior, and larger m improving reconstruction fidelity. The advantage of tuning m as described is that it has a principled interpretation at integer values; namely that of demanding m exact reconstructions from the latent, as derived in Section 3.3.

In this light, KL annealing amounts to starting with $m = \infty$ at the beginning, and smoothly reducing m down to 0 during training. Thus, it is equivalent to optimising the AutoGen lower bound given in Equation 3.10 with varying m during training. However, AutoGen should never require KL annealing.

Scaling of the ELBO is common in multimodal generation, where the reconstruction terms are typically of different orders of magnitude [Vedantam et al., 2018, Wu and Goodman, 2018]. AutoGen can be adapted to provide a bound on a meaningful objective function in multimodal generation with well-scaled terms, by requiring a larger number of reconstructions for one data modality than the other. Autogen thus has broader applications in generative modelling, which we leave to future work.

3.6 Conclusions

In this chapter, we introduced AutoGen: a novel modelling approach to improve the descriptiveness of latent variables in generative models, by combining the log likelihood of a VAE, with the log likelihood of m high-fidelity reconstructions via a stochastic autoencoder. This approach is theoretically principled in that it retains a bound on a meaningful objective, and computationally amounts to a simple factor of $(1 + m)$ in front of the reconstruction term in the standard ELBO. We find that the most natural version of AutoGen (with $m = 1$) provides significantly better reconstructions than the VAE approach to language modelling, and only minimally deprecates generation from the prior.

Chapter 4

Latent Variable Modelling under LDP

The work presented in this chapter was published in Mansbridge et al. [2022].

In the previous chapter, we introduced a new class of latent variable model that facilitated the learning of powerful latent encodings in the presence of a high-capacity generative network.

In this chapter, we consider a novel latent variable model for data privatisation under local differential privacy (LDP). We constrain the approximate posterior distribution such that the addition of carefully calibrated noise induces LDP on the latent manifold. The choice to privatise on the low-dimensional latent manifold allows us to circumvent the 'curse of dimensionality' for LDP, which has plagued the performance of existing privatisation mechanisms when applied to high-dimensional data. The applications of this approach are far reaching; for our empirical experiments we consider the model in the context of data collection methods that protect the privacy of the data owner. We use the collected LDP data to train performant downstream machine learning algorithms. Not only is this a common goal of data collectors, but the performance metrics of these downstream models forms a powerful proxy for measuring the data utility of our privatised training data.

4.1 Introduction

The collection of personal data is ubiquitous, and unavoidable for many in everyday life. The use of such data for training machine learning algorithms has become instrumental in improving the quality and user experience of many products and services.

However, research from Amnesty International [2019] claims that some data collection tactics from large companies constitute a human rights violation. In some instances, the individual may feel the organisation with whom they share their data is trustworthy but despite this, they may still be vulnerable to adversarial third parties. For example, breaches of the health insurance companies Premera and Anthem put at risk the healthcare data of an estimated 11 million and 79 million American citizens respectively [Institute for Critical Infrastructure Technology, 2016], and a recent study suggests that only half of healthcare providers feel capable of defending themselves against cyber-attacks [Martin et al., 2017]. Similarly, the Financial Conduct Authority have revealed accidentally releasing personal details of 1,600 consumers [Jolly, 2020].

Incidents such as these have brought the concept of data privacy into sharp focus, fuelling regulatory changes, as well as a shift in the personal preferences of data owners. There is thus a growing need for data collection methods that preserve individuals' privacy, whilst retaining sufficient data utility for product and service improvement.

Privatising data under *local differential privacy* (LDP) [Kasiviswanathan et al., 2008, Duchi et al., 2013] naturally lends itself to data collection (see Section 2.3.2 for a formal introduction to LDP). As depicted in Figure 2.3, LDP mechanisms allow individuals to privatise their data before sharing it, thus providing a mathematically-provable privacy guarantee for the individual against both a potential adversary and the database administrator.

LDP has its roots in randomised response [Warner, 1965], which preserves the privacy of survey respondents by only having them answer a sensitive binary question truthfully based on the outcome of some binary random variable. For example, a

teacher may be wondering how many students are cheating on a test, but is conscious that students will be unwilling to divulge this incriminating information. Instead, the teacher gives each student a coin to (secretly) flip. If the coin returns tails, the students answer truthfully, and if it returns heads they flip again, answering 'yes' for heads, and 'no' for tails. This mechanism grants a level of plausible deniability to the students – even if they answer 'yes', the teacher knows there is a 1 in 4 chance they didn't cheat. Despite this, the teacher can still use these responses to calculate an estimate of the true rate of cheating. Privatisation is relatively straightforward for this binary problem, but is challenging for more complex data distributions, especially in higher dimensions.

Limited research has gone into developing LDP mechanisms for high-dimensional data, especially those that generalise to different data types. Often dubbed the 'curse of dimensionality', this is a challenging problem in LDP [Zhang et al., 2017, Duchi et al., 2018, Bhowmick et al., 2019].

The local Laplace mechanism (see Definition 2.3.5) is the de facto standard for continuous attributes. Duchi et al. [2018] and Wang et al. [2019] have more recently introduced lower variance continuous mechanisms, though Duchi et al. [2018] emphasise the pessimistic nature of their results in high dimensions. Meanwhile, Wang et al. [2019]'s mechanism induces LDP by collecting $k \ll d$ perturbed attributes per d-dimensional datapoint – experiments in this thesis act on data of dimension $d > 3000$, but Wang et al. [2019]'s mechanism would perturb only $k \leq 4$ attributes – clearly too few to retain sufficient utility in our context. The PrivUnit_2 mechanism [Bhowmick et al., 2019] privatises high-dimensional continuous gradient data for federated learning; however, the authors consider only very high local-ϵ (i.e. low privacy) guarantees, aiming to protect only against accurate data reconstruction rather than arbitrary inferences.

There also exist a number of LDP techniques for specific tasks or data types. Notably, Ding et al. [2017] study the repeated collection of one-dimensional telemetry data for mean and histogram estimation. Erlingsson et al. [2014] develop a technique for collecting aggregate statistics on categorical attributes, with Fanti et al. [2016] extending this to model correlations between dimensions, but neither produce rep-

resentations suitable for downstream learning on high-dimensional data. Ren et al. [2018] discuss the poor performance of the models of Erlingsson et al. [2014] and Fanti et al. [2016] on high dimensional data, and instead estimate the distribution of collected data, from which they sample a synthetic dataset. The range of applications here is limited (see next paragraph), and the approach incurs a high communication cost between the data collector and individuals. In summary, developing a general method for inducing LDP in high dimensions, while preserving data utility, is an open research question.

Central differential privacy (CDP) [Dwork et al., 2006] is a related framework offering protection in an altogether different context. Rather than facilitating the private collection of individual datapoints, CDP mechanisms stop an adversary determining, up to a quantifiable level of certainty, the presence of an individual in a dataset. This is achieved via the calibrated addition of noise to the output of statistical queries on that dataset. However, to achieve this requires the database administrator have access to the full unprivatised dataset. CDP has been used effectively in the related field of private data release. For example, Xie et al. [2018], Triastcyn and Faltings [2019], Acs et al. [2019], Takagi et al. [2021] propose sharing a synthetic dataset composed of samples from generative models trained with a CDP optimisation algorithm [Abadi et al., 2016, Gylberth et al., 2017, Papernot et al., 2017]. While powerful in some scenarios, this approach is not suited to data collection, where we are trying to protect individuals from all external parties, *including* the database administrator. Furthermore, the synthetic data provides no information about the features of specific individuals, and the distribution of the synthetic dataset is static after training the generative model.

In this chapter, we introduce an entirely novel LDP mechanism for private, high-dimensional data collection based on a latent variable model. We motivate this approach based on two observations. First, we note that it is often a good approximation to assume high-dimensional data lives on a much lower-dimensional manifold. Second, the vast majority of organisations collecting personal data already have access to *auxiliary data*; this may be previously collected internal data [Competition and Markets Authority, 2020, Schmidt, 2018], public datasets [Deng et al., 2009, Irvin

et al., 2019, Thomee et al., 2016], or data scraped from the internet, as is commonly used to train unsupervised models [Devlin et al., 2019, Mahajan et al., 2018, Ramesh et al., 2021]. Such auxiliary data can be used to learn this lower-dimensional manifold underlying the data distribution. Thus, to circumvent the pessimistic nature of results for provably optimal LDP mechanisms in high-dimensions [Duchi et al., 2018], we instead propose privatising data on the lower-dimensional manifold, where the negative impact of such results is far smaller.

In this way, we use our auxiliary data to train a latent-variable model [Kingma and Welling, 2014, Rezende et al., 2014] where sampling in latent space adds LDP-inducing, Laplace-distributed noise to the low-dimensional latent. Furthermore, reconstructing a datapoint (i.e. passing it through both the encoder and decoder networks) is equivalent to efficiently adding complex, non-linear noise to the raw features to induce LDP. The addition of this LDP-inducing noise during training ensures that the learnt latents are robust to noise at data privatisation time. We refer to this novel technique as the variational Laplace mechanism (VLM).

In addition, we introduce a novel de-noising approach for downstream model training on data privatised with our mechanisms, enabling the training of performant machine learning models. The training of machine learning models is not only a common goal of data collectors, but downstream model performance forms a powerful proxy for measuring the utility of our LDP training data.

Owing to a wealth of latent variable modelling research on a broad range of data types, including images [Gulrajani et al., 2017], audio [van den Oord et al., 2017], video [Denton and Fergus, 2018], and text [Bowman et al., 2016], the VLM can be easily adapted to many data domains. Finally, the generalisation ability of such models (a) reduces the extent to which auxiliary data must follow the same distribution as the data being collected and (b) allows the collected LDP data to be used for an array of downstream tasks. In particular, we demonstrate that our approach learns powerful LDP data representations, significantly outperforming state-of-the-art LDP benchmark mechanisms on three major applications:

- Privatised data collected with the VLM is used to train downstream machine learning models, demonstrating state-of-the-art utility preservation on high

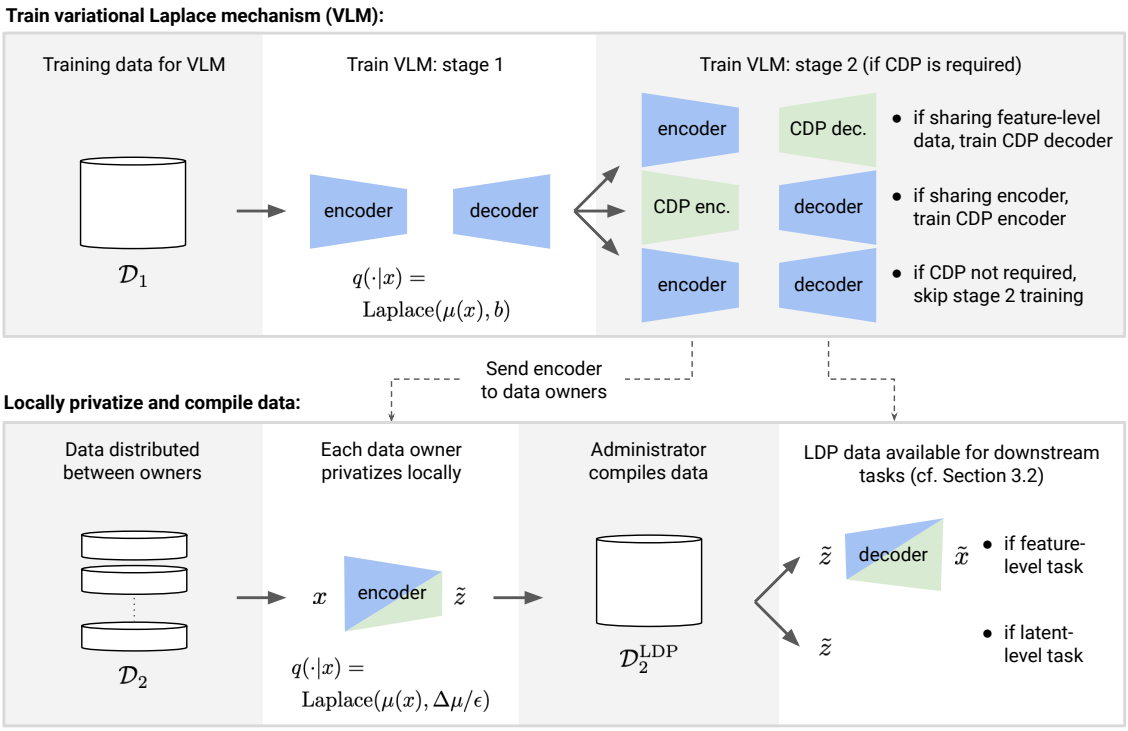


Figure 4.1: Schematic diagram of VLM training (top) and local data privatisation and collection (bottom), as outlined in Section 4.2. Green shading indicates parameters satisfying CDP with respect to the training set.

dimensional data under LDP.

- The VLM can be used, without re-training, to collect distributionally-shifted data for model training. In particular, we train a classifier on privatised data for novel class classification.

- As a mechanism for LDP, the VLM can track the IDs of real individuals whilst privatising their sensitive features. We use this to augment internal data with privatised features from an external source to improve a classifier's performance on the combined feature set.

4.2 Proposed Method

Many existing LDP mechanisms noise each feature independently; by the composition theorem [Dwork and Roth, 2014], the i^{th} feature contributes towards the overall LDP guarantee of the d-dimensional datapoint as $\epsilon = \sum_{i=1}^{d} \epsilon_i$. For fixed ϵ, as d

increases, ϵ_i decreases for each feature i and so the additive noise required to induce ϵ-LDP grows. Equivalently, we can re-frame this concept in terms of sensitivity. Suppose we want to privatise a d-dimensional image $x \in [0,1]^d$ with an additive noise mechanism such as the Laplace mechanism. We clearly see that sensitivity scales with dimension (the ℓ_1 sensitivity is equal to d in this case, c.f. Definition 2.3.6), and thus the additive noise scales with dimension too. In low dimensions, this may not be a cause for concern, but in modern machine learning we often consider datapoints such as images with many thousands of dimensions, resulting in near-total erosion of any information contained within the datapoint.

Furthermore, high-dimensional datapoints like images and large tables often contain highly correlated features. Consequently, noising features independently is wasteful towards privatising the information content.

Even if one does not noise features independently, data utility decreases as dimensionality increases, resulting in poor performance in high dimensions [Duchi et al., 2018]. Instead, we propose a more effective approach: learning an application-agnostic mechanism that privatises a range of data types through the addition of complex, non-linear noise on a learnt, low-dimensional manifold.

To this end, we train a generative latent-variable model to learn a low-dimensional latent representation of our data. The learnt mapping from data space to latent space, under certain constraints, forms our function $f(\cdot)$ in Definition 2.3.7. Crucially, we constrain this manifold such that the sensitivity is fixed and finite, regardless of both the input data dimension and the latent dimension. Under these constraints, the addition of Laplace noise will therefore induce LDP on our latent data representations, giving us a powerful LDP mechanism. Finally, we train the mapping $f(\cdot)$ from data space to the latent manifold such that the induced representations are robust to this LDP-inducing noise, significantly improving utility-retention after privatisation.

In practice, the data collector would share this learnt LDP mechanism with individuals. These individuals would then privatise their data, before sending it to the data collector. In this way, the collector forms a LDP dataset composed of data collected from multiple individuals. This setup is agnostic to choice of downstream-task so can be applied broadly. We consider training downstream machine learning

algorithms as the evaluative task for measuring the utility of these LDP datapoints. Specifically, we use the LDP training data, along with information on the type of noise added, and prior beliefs regarding the distribution of our representation space, to train a classifier network that predicts a label given an input datapoint. This classifier can be trained to act on either clean or privatised datapoints at inference time, depending on the application.

Mechanism Training with Auxiliary Data

Training $f(\cdot)$ requires separate, unlabelled, auxiliary data $\mathcal{D}_1$ with similar distribution to the data $\mathcal{D}_2$ we hope to collect. As discussed in Section 4.1, this is a highly realistic assumption in the current climate.

Access to auxiliary data is widely assumed in many fields of machine learning research. Indeed, the field of transfer learning is largely centred on the idea that knowledge gained from solving one problem (through model training on a given "auxiliary" dataset) can be used to aid learning in another problem. Notably, vast bodies of research in fields like computer vision and natural language processing have utilised auxiliary data through the use of pre-trained embeddings [Devlin et al., 2019, Krizhevsky, 2009] to solve entirely separate tasks [Lee et al., 2019, Chen et al., 2020].

For experiments in this chapter, we simulate a scenario in which we have access to auxiliary data by splitting our raw training data into two sets: the auxiliary data $\mathcal{D}_1$ used for mechanism training and a dataset $\mathcal{D}_2$ that we wish to privatise and collect under LDP. We then use the privatised version of $\mathcal{D}_2$ to train downstream models. In some experiments, we choose this data split such that $\mathcal{D}_1$ and $\mathcal{D}_2$ follow the same distribution, while in other experiments we split the data such that $\mathcal{D}_1$ and $\mathcal{D}_2$ follow significantly different distributions. Crucially, in neither scenario is the data to be privatised contained in the mechanism training data.

In some instances, the auxiliary data is assumed to be sensitive and thus we train the encoder $f(\cdot)$ of our mechanism under CDP. This protects the privacy of the members of $\mathcal{D}_1$, since we share $f(\cdot)$ with members of $\mathcal{D}_2$ during collection.

4.2.1 Variational Laplace Mechanism (VLM)

We assume each datapoint x is generated by a random process involving a latent variable z of dimension d. We then optimise a lower bound on the log likelihood [Kingma and Welling, 2014]

$$\log p(x) \geq \mathbb{E}_{q_\phi(z|x)} \left[\log p_\theta(x|z)\right] - D_{\mathrm{KL}}\left[q_\phi(z|x)||p(z)\right] \tag{4.1}$$

where $p(z)$ is the prior and $q_\phi(z|x)$ is the approximate posterior over the latent representation. The generative distribution $p_\theta(x|z)$ and approximate inference distribution $q_\phi(z|x)$ are parameterised by neural networks, with learnable parameters θ and ϕ respectively. Since we aim to learn a local Laplace mechanism, we choose

$$p(z) = \prod_{i=1}^{d} p(z_i) \quad \text{and} \quad q_\phi(z|x) = \prod_{i=1}^{d} q_\phi(z_i|x) \tag{4.2}$$

where $p(z_i) = \mathrm{Laplace}(0, 1/\sqrt{2})$ and $q_\phi(z_i|x) = \mathrm{Laplace}(f_\phi(x)_i, b)$.

We parameterise $f_\phi(\cdot)$ with a neural network and restrict its output via a carefully chosen activation function $\nu(\cdot)$ acting on the final layer $f_\phi(\cdot) = \nu(h_\phi(\cdot))$, where

$$\nu(h) = h * \min\{1, l/||h||_1\} \tag{4.3}$$

for some $l > 0$. This ensures all outputs of $f_\phi(\cdot)$ lie within an ℓ_1-norm l of the origin, and so $\Delta f_\phi = 2l$. We fix the scale of the Laplace distribution to $b = 2l/\epsilon_x$,

Proposition 4.2.1. *A sample $\tilde{z}$ drawn from the encoder distribution $q_\phi(z|x)$ forms a representation of x that satisfies ϵ_x-LDP.*

Proof. Drawing a sample from $q_\phi(z|x)$ is equivalent to passing a point x through the mapping $f_\phi(\cdot)$, before adding $\mathrm{Laplace}(0, 2l/\epsilon_x)$ noise to each dimension. Since the sensitivity of $f_\phi(\cdot)$ is $\Delta f_\phi = 2l$, this is equivalent to passing x through the local Laplace mechanism $\mathcal{M}\left(x, f_\phi(\cdot), \epsilon_x\right)$ from Definition 2.3.7. Thus, $\tilde{z}$ is a representation of x that satisfies ϵ_x-LDP. □

While b is fixed at data privatisation time, we experiment with different values of b during mechanism training. Experiments conducted in Section 4.3.1.3, in which we

privatise data using other dimensionality reduction techniques, strongly suggest that this addition of noise during training is important to learning a latent representation that is robust to the ϵ_x-LDP noise requirements at privatisation time.

We note also that the post-processing property of LDP applies to these privatised latent representations:

Proposition 4.2.2. *If a point in latent space satisfies ϵ-LDP, then this point still satisfies ϵ-LDP after being passed through a deterministic function, such as the function that parameterises the mean of the decoder network.*

Proof. We follow an approach similar to the proof that central differential privacy is immune to post-processing [Dwork and Roth, 2014]. Let $\mathcal{M} : \mathcal{X} \to \mathcal{Z}$ be a randomised algorithm that satisfies ϵ-LDP and $g : \mathcal{Z} \to \mathcal{Z}'$ be an arbitrary deterministic mapping. Let $S \subseteq \mathcal{Z}'$ and $T = \{z \in \mathcal{Z} : g(z) \in S\}$. Then

$$p\big(g(\mathcal{M}(x)) \in S\big) = p\big(\mathcal{M}(x) \in T\big) \tag{4.4}$$

$$\leq e^{\epsilon}\, p\big(\mathcal{M}(x') \in T\big) \tag{4.5}$$

$$= e^{\epsilon}\, p\big(g(\mathcal{M}(x')) \in S\big) \tag{4.6}$$

□

This means that a reconstruction $\tilde{x}$ obtained by passing $\tilde{z}$ through the decoder network $p_\theta(\cdot|z)$ also satisfies ϵ_x-LDP. This facilitates the collection of privatised datapoints on either representation level $\tilde{z}$, or original-feature level $\tilde{x}$, depending on the data collector's preference. We refer to this method as the Variational Laplace Mechanism (VLM).

In order to collect data we must share the VLM encoder $f_\phi(\cdot)$ with the (potentially untrusted) data owner. The parameters of $f_\phi(\cdot)$ may contain information about members of the auxiliary dataset $\mathcal{D}_1$, as discussed in Section 2.3.1.1. Therefore, in the scenario that $\mathcal{D}_1$ is a sensitive or non-public dataset, the parameters of the encoder may need to satisfy CDP with respect to $\mathcal{D}_1$. To achieve this CDP guarantee, we found the following two-stage training approach to be effective:

- **Stage 1:** Train a VLM with encoding distribution $q_\phi(z|x)$ and decoding distribution $p_\theta(x|z)$ using a non-CDP optimisation algorithm. In this work, we use Adam [Kingma and Ba, 2015].

- **Stage 2:** Fix θ and re-initialise the encoder with a new distribution $q_{\phi_{\text{private}}}(z|x)$. Optimise ϕ_{private} using a CDP optimisation algorithm. In this work, we use DP-Adam [Gylberth et al., 2017]. See Section 2.3.1.1 for an overview of CDP optimisation algorithms.

Choice of latent LDP mechanism: The only restriction on our choice of privatisation mechanism is that the noise distribution must be reparameterisable (see Section 2.1.1.2). Consequently, mechanisms such as the Gaussian mechanism could be used by changing the activation $\nu(\cdot)$, and the choice of distribution over latent space. We focus on the Laplace mechanism since it is well-studied in the LDP literature and has stronger guarantees than the Gaussian mechanism which satisfies only $(\epsilon, \delta > 0)$-LDP (see e.g. Wang et al. [2021]). While work has been done on constructing minimax-optimal mechanisms (see e.g. Duchi et al. [2018], Bhowmick et al. [2019]), we emphasise that the compelling performance of the VLM is largely afforded by the mapping of data, via $f_\phi(\cdot)$, to powerful, low-dimensional representations of our high-dimensional data, which are easier to privatise than the raw data. Furthermore, the minimax-optimality guarantees are generally given in the context of simpler tasks such as mean estimation. We do not restrict our representations to these limited downstream tasks; notably, in this work we learn the parameters of (non-linear) neural network classifiers.

4.2.2 Collecting LDP labels

A key requirement for the supervised training of downstream machine learning models is access to labelled training datapoints. Having introduced a powerful approach for collecting high-dimensional data under LDP, we now introduce an approach for collecting corresponding labels under LDP. We demonstrate the capabilities of the VLM empirically in the context of training classifiers, and so we collect and privatise

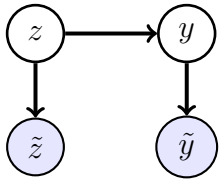


Figure 4.2: Graphical model representing the dependency structure between datapoints z, labels y and their corresponding LDP versions $\tilde{z}$ and $\tilde{y}$. The blue shading indicates that the random variable is observed.

a discrete scalar class label $y \in \{1, \ldots, K\}$ alongside our privatised representation $\tilde{z}$ (or $\tilde{x}$). To obtain a privatised label $\tilde{y}$, we flip y with probability

$$p(\tilde{y} = i | y = j) = \frac{e^{\epsilon_y} \mathbb{I}(i = j)}{e^{\epsilon_y} + K - 1} + \frac{\mathbb{I}(i \neq j)}{e^{\epsilon_y} + K - 1} \tag{4.7}$$

which induces ϵ_y-LDP (see Wang et al. [2014] for proof). If $y \in \mathbb{R}$ (e.g. for regression) one could instead privatise this with, say, a local Laplace Mechanism. By the composition theorem [Dwork and Roth, 2014], the tuples $(\tilde{z}, \tilde{y})$ or $(\tilde{x}, \tilde{y})$ satisfy ϵ-LDP, where $\epsilon = \epsilon_x + \epsilon_y$. Downstream models may be more robust to label noise than feature noise, or vice versa, so for fixed ϵ we set $\epsilon_x = \lambda\epsilon$ and $\epsilon_y = (1 - \lambda)\epsilon$, with λ chosen to optimise downstream model performance.

4.2.3 Downstream Model Training on LDP Data

We introduce a procedure for training machine learning models given access only to the collected ϵ-LDP labelled training dataset privatised with the VLM. This is a primary motive for data collectors, and furthermore, downstream model performance provides a powerful proxy for measuring the utility in our collected LDP data. Specifically, we learn a classifier that predicts the underlying clean target variable given either a clean datapoint or a privatised datapoint, depending on the application. We treat the unobserved underlying clean features as latent variables that we marginalise out in our training objective. Figure 4.2 outlines the dependency structure between the privatised features $\tilde{z}$, privatised labels $\tilde{y}$, and the underlying clean features z and clean labels y.

First, we assume the VLM training data $\mathcal{D}_1 := \{\hat{x}_m\}_{m=1}^M$ (which we do have access to) follows a similar distribution to data $\mathcal{D}_2$ being collected, and place an

empirical prior $p(z|\mathcal{D}_1) = \frac{1}{M}\sum_{m=1}^{M}\delta(z-\hat{z}_m)$ over the (clean) representation space z of our collected data $\mathcal{D}_2$, where $\hat{z}_m = f_\phi(\hat{x}_m)$. We then marginalise out both label and representation noise, and maximise

$$p(\tilde{y},\tilde{z}|\mathcal{D}_1) = \int p(\tilde{y}|z)p(\tilde{z}|z)p(z|\mathcal{D}_1)\,dz \tag{4.8}$$

$$= \frac{1}{M}\sum_{m=1}^{M}\int p(\tilde{y}|z)p(\tilde{z}|z)\delta(z-\hat{z}_m)\,dz \tag{4.9}$$

$$= \frac{1}{M}\sum_{m=1}^{M} p(\tilde{y}|\hat{z}_m)p(\tilde{z}|\hat{z}_m) \tag{4.10}$$

$$= \frac{1}{M}\sum_{m=1}^{M}\sum_{y} p(\tilde{y}|y)p_\psi(y|\hat{z}_m)p(\tilde{z}|\hat{z}_m) \tag{4.11}$$

We can use $p_\psi(y|z)$ to classify *clean* (i.e. non-privatised) latents at inference time. To classify *privatised* latents, we only marginalise label noise and train $p_\xi(y|\tilde{z})$ by maximising:

$$p(\tilde{y}|\tilde{z}) = \sum_{y} p(\tilde{y}|y)p_\xi(y|\tilde{z}) \tag{4.12}$$

We find experimentally that $p_\xi(y|\tilde{z})$ can also achieve a high accuracy classifying *clean* latents. Though it is generally outperformed by $p_\psi(y|z)$ for this task, it can achieve better performance in scenarios where we don't have access to a good prior $p(z|\mathcal{D}_1)$. This is particularly evident in the novel class classification task, where $\mathcal{D}_1$ and $\mathcal{D}_2$ come from significantly different distributions, as we discover experimentally in Sections 4.3.2 and 5.3.2.

The classification of both clean and private data are challenging problems with a multitude of applications. To our knowledge, no existing work has achieved compelling results on either problem in the high-dimensional setting. Figure 4.1 outlines a schematic of mechanism training, data privatisation, collection, and classifier training.

4.2.4 Hyperparameter Tuning Under LDP

Typically for model validation and testing, one needs access to clean labels y (and clean data x when validating a classifier acting on clean data at inference time). The data collector does not have access to this when collecting under LDP. However we note they need only collect privatised model performance metrics on validation/ test sets, rather than directly accessing clean datapoints.

To do this, the trained classifier is sent to members of a validation/ test group, who would determine whether the classifier was correct $c \in \{0, 1\}$ on their data. Validation set members then return an ϵ_c-LDP version of c which we denote $\tilde{c} \in \{0, 1\}$, flipped with probability $p = 1/(e^{\epsilon_c} + 1)$. The true validation set accuracy $A = \frac{1}{N_{\text{val}}} \sum_{n=1}^{N_{\text{val}}} c_n$ can be estimated from the privatised accuracy $\tilde{A} = \frac{1}{N_{\text{val}}} \sum_{n=1}^{N_{\text{val}}} \tilde{c}_n$ using $A = (\tilde{A} - p)/(1 - 2p)$ [Warner, 1965]. We use this method when conducting a grid search over hyperparameters of our model, and to determine when to stop training.

4.3 Applications and Experiments

The fundamental goal of a LDP mechanism is to maximise the retention of data utility while guaranteeing ϵ-LDP. There are many ways to measure utility; we train downstream models on our LDP data, and use downstream model performance metrics as a proxy to measure the utility retained in our LDP training data. We further demonstrate the versatility of our method by outlining a non-exhaustive list of real-world-inspired applications, with corresponding experiments carried out on MNIST [LeCun et al., 1998] and Lending Club[1] – a tabular, binary classification task. These tasks represent a step up in difficulty relative to those previously studied in the LDP data collection literature. Full details of the experimental setup for these tasks can be found in Appendix A.

In Sections 4.3.1 and 4.3.2, we investigate the task of training a classifier on LDP data in order to classify *clean* (i.e. non-privatised) datapoints at inference time. In Section 4.4, we train a classifier on LDP data that classifies LDP datapoints

[1] `https://www.kaggle.com/wordsforthewise/lending-club` (License CC0: Public Domain)

at inference time, and refer to the performance metric of this separate use case as *private* classification accuracy. Unless otherwise stated, we collect and train the classifier using privatised data $\tilde{z}$ on representation level, rather than privatised data $\tilde{x}$ on feature level (obtained by passing $\tilde{z}$ through the decoder as in Proposition 4.2.2). We optimise hyperparameters under LDP using a private grid search, as outlined in Section 4.2.4. Final hyperparameter values are given in Appendix A.4.

Benchmarks: We benchmark results against the Laplace mechanism, Duchi's mechanism [Duchi et al., 2018] and the PrivUnit$_2$/ ScalarDP mechanism [Bhowmick et al., 2019]. To our knowledge, the latter two represent current state-of-the-art. See Appendix A.3 for implementation details. We stress that classifiers trained with DP-SGD [Abadi et al., 2016] do not constitute meaningful benchmarks since they provide no mechanism for LDP data collection, and require access to a labelled, non-LDP classifier training set. While we assume access to non-LDP auxiliary data, this does not need to be labelled, nor from the same distribution as our collected training data, as we demonstrate in Section 4.3.2.

4.3.1 Data Collection

A fundamental objective of this work is for organisations to utilise clean auxiliary data to significantly improve the future, *private* collection of user data. Vast numbers of organisations collecting data have access to existing datasets already. For example, public health bodies have access to medical images, tech companies have access to user data, and multinationals may have access to certain data from users in some regions but not others. In addition, there exists a broad array of public datasets designed specifically for training machine learning models, whilst scraping data from the internet has also become commonplace for training unsupervised models. This existing 'auxiliary' data can be used to train mechanisms that facilitate future *private* data collection by organisations; this may be data from a group of patients in a particular study, e-commerce data from a broader group of users, or multinational data from a region where legislation is stricter, and so only private collection is acceptable.

Table 4.1: Accuracy of classifiers trained on data collected using different LDP mechanisms. Each row shows the ϵ-LDP guarantee for the collected training set. Error bars represent ±1 standard deviation from the mean over 3 trials.

	Privacy Level	VLM	VLM ($\epsilon_{\text{CDP}} = 5$)	VLM ($\epsilon_{\text{CDP}} = 1$)	PrivUnit	Duchi	Laplace
MNIST	$\epsilon_{\text{LDP}} = 10$	**86.1±1.0**	78.6±1.2	72.6±1.4	38.2±4.6	13.9±4.5	9.2±1.4
	$\epsilon_{\text{LDP}} = 8$	**82.1±1.6**	75.9±0.9	70.3±1.9	15.1±4.2	14.3±5.0	9.8±2.8
	$\epsilon_{\text{LDP}} = 6$	**72.8±3.2**	66.2±2.4	60.3±2.6	12.4±1.6	13.3±3.8	10.6±0.3
	$\epsilon_{\text{LDP}} = 4$	**61.3±2.8**	50.7±2.5	46.9±3.3	9.2±2.6	14.1±4.9	10.0±0.6
	$\epsilon_{\text{LDP}} = 2$	**35.3±9.7**	16.8±4.4	17.1±3.2	9.5±2.9	14.0±5.6	9.0±1.1
	$\epsilon_{\text{LDP}} = 1$	16.9±1.3	**17.2±1.3**	14.4±4.3	11.3±4.5	10.0±1.1	10.0±0.8
	No Privacy	94.9±0.2	87.2±0.4	83.7±0.5	96.0±0.4	96.0±0.4	96.0±0.4

	Privacy Level	VLM	VLM ($\epsilon_{\text{CDP}} = 5$)	VLM ($\epsilon_{\text{CDP}} = 1$)	Duchi	Laplace
Lending Club	$\epsilon_{\text{LDP}} = 10$	**63.7±0.6**	**63.7±0.4**	63.2±0.5	56.0±5.5	50.1±0.6
	$\epsilon_{\text{LDP}} = 8$	63.2±0.3	**63.4±0.2**	63.1±0.4	53.0±4.2	50.3±0.8
	$\epsilon_{\text{LDP}} = 6$	**62.6±0.5**	**62.6±0.5**	**62.6±0.6**	52.7±2.9	49.6±1.1
	$\epsilon_{\text{LDP}} = 4$	61.1±1.2	61.3±1.4	**61.5±1.5**	50.1±1.4	49.7±1.2
	$\epsilon_{\text{LDP}} = 2$	**53.9±4.3**	53.6±4.1	**53.9±2.4**	50.1±0.9	49.9±0.7
	$\epsilon_{\text{LDP}} = 1$	**55.1±4.5**	54.9±4.5	54.6±4.1	49.3±2.5	49.5±0.9
	No Privacy	65.0±0.3	65.1±0.3	64.6±0.2	65.7±0.2	65.7±0.2

In this section, we run experiments on MNIST and Lending Club. As outlined in Appendix A.2, we split the data such that 75% forms the 'auxiliary' dataset $\mathcal{D}_1$ used to train the VLM, and 25% forms an external dataset $\mathcal{D}_2$ that we privatise under ϵ-LDP. We split the dataset in this manner to simulate a real-world scenario but emphasise that in practice, the size of $\mathcal{D}_1$ and $\mathcal{D}_2$ would be pre-determined by the amount of auxiliary data the organisation has access to, and the amount of data they are able to collect, respectively.

Since a key motivation for organisations collecting data is to train machine learning models, we test the utility of the privatised $\mathcal{D}_2$ by using it to train a classifier. We compare the performance of the classifiers trained on data $\mathcal{D}_2$ privatised using the VLM, against classifiers trained on data $\mathcal{D}_2$ privatised with benchmark LDP mechanisms. We train three VLM mechanisms for each task. The first assumes $\mathcal{D}_1$ is

a non-sensitive dataset and so does not train the encoder to satisfy $(\epsilon_{\text{CDP}}, \delta_{\text{CDP}})$-CDP guarantees. The remaining mechanisms guarantee privacy for individuals from $\mathcal{D}_1$ at the $\epsilon_{\text{CDP}} = \{1, 5\}$ level, and this is done using the two-stage approach discussed in Section 4.2.1. For all stated $(\epsilon_{\text{CDP}}, \delta_{\text{CDP}})$-CDP guarantees we use $\delta_{\text{CDP}} = 10^{-5}$. Results are shown in Table 4.1.

The results conclusively demonstrate that classifiers trained on ϵ-LDP data privatised with the VLM significantly outperform classifiers trained on the benchmarks at every local ϵ tested, indicating our methods facilitate much greater retention of utility. At lower local ϵ values, we are reducing the data utility of our classifier training set in exchange for stronger privacy guarantees, and the classifier performance drops as expected. However, even at local $\epsilon = 1$, we are achieving roughly the same performance as PrivUnit, the top performing benchmark, at local $\epsilon = 8$.

For MNIST, we see that classifier accuracy deteriorates slightly when data is collected using mechanisms trained with lower ϵ_{CDP} guarantees. This suggests that slightly less information is contained in representations privatised with LDP mechanisms satisfying the stricter CDP guarantees. This is to be expected, since in order to train the VLM encoder under CDP we must add noise to our gradients, hindering our ability to find a good optimum. Despite this, even the VLM satisfying the most stringent CDP guarantee significantly outperforms all benchmarks at all ϵ_{LDP} values.

For Lending Club, we see virtually no deterioration in classification accuracy when collecting data using a CDP encoder. We attribute this to the larger training set $\mathcal{D}_1$. A larger training set means that, for a fixed number of training iterations and fixed batch size, a single datapoint in $\mathcal{D}_1$ contributes fewer times to gradient updates and so less noise needs to be added to each gradient to guarantee the $(\epsilon_{\text{CDP}}, \delta_{\text{CDP}})$-CDP requirement with respect to $\mathcal{D}_1$.

The "no privacy" column indicates performance when no noise is added; for the benchmarks, this means training the classifier directly on non-private data $\mathcal{D}_2$, whilst for the VLM, it means training it on (un-noised) latent representations of datapoints in $\mathcal{D}_2$.

Table 4.2: Accuracy of classifiers trained on either feature-level data or representation-level data collected with the VLM. Each row shows the ϵ-LDP guarantee for the collected training set. Error bars represent ±1 standard deviation from the mean over 3 trials.

	Privacy Level	VLM (representation level)	VLM (feature level)	PrivUnit	Duchi	Laplace
MNIST	$\epsilon_{\text{LDP}} = 10$	**86.1±1.0**	70.2±3.8	38.2±4.6	13.9±4.5	9.2±1.4
	$\epsilon_{\text{LDP}} = 8$	**82.1±1.6**	64.4±4.9	15.1±4.2	14.3±5.0	9.8±2.8
	$\epsilon_{\text{LDP}} = 6$	**72.8±3.2**	50.5±3.4	12.4±1.6	13.3±3.8	10.6±0.3
	$\epsilon_{\text{LDP}} = 4$	**61.3±2.8**	45.6±0.9	9.2±2.6	14.1±4.9	10.0±0.6
	$\epsilon_{\text{LDP}} = 2$	**35.3±9.7**	25.7±6.9	9.5±2.9	14.0±5.6	9.0±1.1
	$\epsilon_{\text{LDP}} = 1$	16.9±1.3	**19.5±9.1**	11.3±4.5	10.0±1.1	10.0±0.8
	No Privacy	94.9±0.2	90.4±0.7	96.0±0.4	96.0±0.4	96.0±0.4

	Privacy Level	VLM (representation level)	VLM (feature level)	Duchi	Laplace
Lending Club	$\epsilon_{\text{LDP}} = 10$	**63.7±0.6**	63.4±0.2	56.0±5.5	50.1±0.6
	$\epsilon_{\text{LDP}} = 8$	63.2±0.3	**63.3±1.0**	53.0±4.2	50.3±0.8
	$\epsilon_{\text{LDP}} = 6$	62.6±0.5	**63.4±0.4**	52.7±2.9	49.6±1.1
	$\epsilon_{\text{LDP}} = 4$	61.1±1.2	**62.1±0.4**	50.1±1.4	49.7±1.2
	$\epsilon_{\text{LDP}} = 2$	53.9±4.3	**56.3±2.0**	50.1±0.9	49.9±0.7
	$\epsilon_{\text{LDP}} = 1$	**55.1±4.5**	52.7±3.3	49.3±2.5	49.5±0.9
	No Privacy	65.0±0.3	65.0±0.2	65.7±0.2	65.7±0.2

4.3.1.1 Feature level collection

In the previous experiments, the classifiers were trained on LDP data collected on latent representation level. In other words, data owners passed their data x through the VLM to obtain privatised latent representations $\tilde{z}$, and their labels through a flip mechanism to obtain private labels $\tilde{y}$. In certain scenarios, the data collector may wish to collect data on original feature level. The data collector can achieve this by passing the privatised latent representations $\tilde{z}$ through the decoder network; the resulting feature-level datapoints $\tilde{x}$ still satisfy the LDP guarantees, as proved in Proposition 4.2.2. This might be particularly relevant to tabular data, where the collector may wish to run different queries on individual features, or simply store the privatised data in the same format as the underlying non-private data.

When the LDP classifier training set $(\tilde{x}, \tilde{y})$ is on feature level, we can no longer use the classifier objective in Equation 4.11. This would entail replacing the zero-mean Laplace noise $p(\tilde{z}|z)$ with the feature-level noise distribution $p(\tilde{x}|x)$, which has no closed-form solution. Instead, we optimise our classifier with the objective from Equation 4.12.

Results, given in Table 4.2, show the MNIST classifier performs better when trained on representation-level data than feature-level data. We attribute this to both the simpler classifier, and the performance of the decoder, which has to learn to reconstruct high-dimensional datapoints from low-dimensional latents that have been subject to large quantities of Laplace noise. This is an extremely challenging problem. Despite this, the feature-level classifier still significantly outperforms all benchmarks. For Lending Club, we see little change between the feature-level and representation-level classifiers. We hypothesise that the lower-dimensional datapoints are easier for the VLM to reconstruct than MNIST images, and so we do not see the same drop in performance.

4.3.1.2 Effects of Reducing Auxiliary Dataset Size

As previously discussed, it is realistic to assume most data collectors have access to auxiliary data. However, the volume of data they have access to may vary significantly depending on the data type, data collector, and data owner. For the previous experiments, we used $\eta = 75\%$ of the training data for the VLM training set $\mathcal{D}_1$, and privatise the remaining 25%, to form the classifier training set $\mathcal{D}_2$. In this ablation study we re-run these experiments, but train the VLM on smaller proportions $\eta = \{50\%, 25\%, 10\%\}$ of the MNIST training set, corresponding to 30,000, 15,000, and 6,000 unlabelled training images respectively.

For the benchmark mechanisms, we experiment with varying the proportion of the MNIST training set assigned to $\mathcal{D}_2$, to determine whether a large classifier training set could allow the benchmarks to compete with the VLM. Since the benchmark mechanisms do not require a pre-training set $\mathcal{D}_1$, we consider the most favourable scenario in which 100% of the data (60,000 labelled images) is assigned to the privatised classifier training set $\mathcal{D}_2$.

Table 4.3: Accuracy of classifiers trained on data collected using different LDP mechanisms. η represents the proportion of the MNIST training set used for VLM training. Each row shows the ϵ-LDP guarantee for the collected training set. Error bars represent ± 1 standard deviation from the mean over 3 trials.

Privacy Level	VLM ($\eta = .75$)	VLM ($\eta = .50$)	VLM ($\eta = .25$)	VLM ($\eta = .10$)	PrivUnit	Duchi
$\epsilon_{\text{LDP}} = 10$	86.1±1.0	**86.4±0.2**	85.1±0.2	81.1±1.8	47.3±2.3	26.3±3.0
$\epsilon_{\text{LDP}} = 8$	82.1±1.6	**84.0±1.0**	81.9±0.5	77.4±1.7	29.2±5.5	20.5±0.8
$\epsilon_{\text{LDP}} = 6$	72.8±3.2	**78.0±3.1**	75.4±2.7	67.9±1.4	16.2±2.7	20.4±2.9
$\epsilon_{\text{LDP}} = 4$	61.3±2.8	**61.6±2.3**	60.5±3.5	57.6±3.5	12.0±3.7	17.0±2.7
$\epsilon_{\text{LDP}} = 2$	35.3±9.7	35.2±1.7	**44.8±4.0**	37.9±1.4	10.7±0.7	14.0±2.7
$\epsilon_{\text{LDP}} = 1$	16.9±1.3	14.1±1.9	16.3±0.8	**18.0±2.7**	14.3±0.4	13.9±1.0

Results, shown in Table 4.3, show the accuracy of classifiers trained on data collected with each mechanism at a range of ϵ-LDP guarantees and a range of data splits. We see that the increased size of $\mathcal{D}_2$ has improved benchmark performance (benchmark results for classifiers trained using only 25% of the MNIST training set are given in Table 4.1). Despite this, all of the VLMs significantly outperform each benchmark mechanism at every ϵ-LDP guarantee tested. We omit the Laplace mechanism results due to space constraints, but note that it underperforms both PrivUnit and Duchi's mechanism at every ϵ value tested. We note that these VLM mechanisms use hyperparameters optimised for experiments in Section 4.3 where the VLM used $\eta = 75\%$ of the training data, and so performance on smaller VLM training sets could potentially be improved with further hyperparameter tuning.

The goal of this ablation study was to see how the performance of the mechanisms is affected by significant changes in the number of training points. We emphasise that in real-world scenarios, the size of $\mathcal{D}_1$ is limited by the quantity of auxiliary data the collector has access to, and the size of $\mathcal{D}_2$ is limited by the quantity they are willing or able to collect. Crucially, throughout these experiments, we always maintain $\mathcal{D}_2 \cap \mathcal{D}_2 = \varnothing$, as is reflective of real-world applications.

Table 4.4: Accuracy of classifiers trained on LDP data, collected using either a PCA-based LDP mechanism or a VLM (with either linear or non-linear encoder-decoder network architectures). Each row shows the ϵ-LDP guarantee for the collected training set. Error bars represent ± 1 standard deviation from the mean over 3 trials.

	Privacy Level	VLM	Linear VLM	PCA
MNIST	$\epsilon_{\text{LDP}} = 10$	**86.1±1.0**	54.1±3.2	17.3±8.1
	$\epsilon_{\text{LDP}} = 8$	**82.1±1.6**	31.3±2.2	15.1±4.9
	$\epsilon_{\text{LDP}} = 6$	**72.8±3.2**	25.1±1.8	15.0±5.4
	$\epsilon_{\text{LDP}} = 4$	**61.3±2.8**	22.5±3.2	14.8±4.9
	$\epsilon_{\text{LDP}} = 2$	**35.3±9.7**	10.5±1.7	15.1±5.6
	$\epsilon_{\text{LDP}} = 1$	**16.9±1.3**	10.3±1.7	15.7±6.6
	No Privacy	94.9±0.2	85.9±0.2	86.4±0.3

4.3.1.3 Comparison with Simpler Dimensionality-Reduction Approaches

In this section, we aim to determine whether a comparable level of data utility can be achieved for points privatised with Laplace mechanisms based on simpler dimensionality-reduction techniques than the VLM. Specifically, we compare the standard VLM against a VLM parameterised with linear encoder and decoder networks, and to a PCA-based local Laplace mechanism.

The experimental set-up for PCA is the same as in the data collection experiment: we use the dataset $\mathcal{D}_1$ to learn the principal components, and use these to reduce our dataset $\mathcal{D}_2$ to dimension 8, as in the MNIST experiments for the VLM. We then add Laplace noise to privatise $\mathcal{D}_2$, and this privatised dataset is used to train a classifier.

There are 2 fundamental differences between these models. Firstly, both the PCA mechanism and the linear VLM use linear mappings $f_\phi(\cdot)$ from data space to representation space, while the standard VLM uses a non-linear mapping parameterised by a deep neural network. Secondly, both the standard VLM and linear VLM add Laplace noise during training, and so we expect the representations to be more robust to privatisation noise than the data reduced with PCA.

Results are shown in Table 4.4. Firstly, we see that when no noise is added to the representations (i.e. no privacy is induced), the PCA and linear-VLM mechanisms are outperformed by the VLM, suggesting the clean 'linear' representations contain less information than the VLM representations for downstream model training.

Table 4.5: Accuracy of classifiers for novel class classification, trained on data collected using different LDP mechanisms. Each row shows the ϵ-LDP guarantee for the collected training set. Error bars represent ± 1 standard deviation from the mean over 3 trials.

	Privacy Level	VLM	PrivUnit	Duchi	Laplace
MNIST	$\epsilon_{\text{LDP}} = 10$	**84.0±0.3**	77.6±8.1	69.7±1.6	47.0±4.4
	$\epsilon_{\text{LDP}} = 8$	**80.9±1.8**	71.0±2.6	68.7±3.1	47.3±4.0
	$\epsilon_{\text{LDP}} = 6$	**82.5±0.5**	75.3±4.2	70.5±2.5	48.5±2.1
	$\epsilon_{\text{LDP}} = 4$	**80.7±0.3**	54.5±3.9	66.5±6.5	48.7±2.0
	$\epsilon_{\text{LDP}} = 2$	**72.4±0.5**	54.4±8.3	58.9±8.6	49.6±1.0
	$\epsilon_{\text{LDP}} = 1$	**55.8±8.7**	54.6±10.1	50.4±0.2	48.5±2.2
	No Privacy	94.0±0.4	97.8±0.3	97.8±0.3	97.8±0.3

Secondly, we see that performance drops significantly when any privacy-inducing noise is added to the PCA representations, whilst for both linear and non-linear VLMs, the addition of noise has less impact on classifier accuracy. Indeed at $\epsilon = 10$, PCA is outperformed by the PrivUnit benchmark – a technique that does not use dimensionality reduction (for benchmark results, see Table 4.1). This suggests that the strong results for the VLM are not simply attributable to dimensionality reduction, but to the VLM's ability to learn powerful low-dimensional representations that are robust to privacy-inducing noise, and well-suited to downstream model training.

All experiments so far have assumed $\mathcal{D}_1$ and $\mathcal{D}_2$ come from similar distributions, though in practice the distributions may differ. For example, $\mathcal{D}_2$ could be sales data collected in a different time period to $\mathcal{D}_1$, or user data collected in a different region. In Section 4.3.2, we study the extreme scenario of distributional shift, in which $\mathcal{D}_2$ contains classes unseen in $\mathcal{D}_1$.

4.3.2 Novel-Class Classification

As discussed in Section 4.3.1, the auxiliary data $\mathcal{D}_1$ and the data to be collected $\mathcal{D}_2$ may follow different data distributions. In one extreme case, the desired task on $\mathcal{D}_2$ may be to predict membership in a class that is not even present in dataset $\mathcal{D}_1$. For example, in a medical application there may be a large existing dataset of chest scans

$\mathcal{D}_1$, but a public health body may want to collect data $\mathcal{D}_2$ from patients with a novel disease in order to train a novel-disease classifier to distribute to hospitals. Similarly, a software developer may have access to an existing dataset $\mathcal{D}_1$, but want to predict software usage data for $\mathcal{D}_2$, whose label is specific to the UI of a new release.

We run this experiment on MNIST, where the auxiliary $\mathcal{D}_1$ contains training images from classes 0 to 8, (with a small number of images held out for classifier training), and $\mathcal{D}_2$ contains all training images from class 9. Full implementation details are given in Appendix A.2. As in Section 4.2, we first train the VLM on $\mathcal{D}_1$, then privatise all images in $\mathcal{D}_2$ (we do not collect labels since all collected images have the same label). We then train a binary classifier on the dataset formed of the private 9's and the held out auxiliary images from classes 0-8 (which we privatise and label 'not 9's'). Since $\mathcal{D}_1$ contains no datapoints from class 9, the prior from Equation 4.11 is no longer as accurate. Consequently, as discussed in Section 4.2.3, we found training our classifier by maximising $\log p(y|\tilde{z})$ directly led to better performance.

Results are shown in Table 4.5. We see that PrivUnit and Duchi's mechanism both perform significantly better on this simpler binary classification task than in the 10-class classification problem of Section 4.3.1. Despite this, the VLM still outperforms the benchmarks at all ϵ values tested, once again indicating a much greater retention of data utility than is achieved by the benchmark mechanisms.

4.3.3 Data Joining

An organisation training a classifier on some labelled dataset $\mathcal{D}_1$ could potentially improve performance by augmenting their dataset with other informative features, and so may want to join $\mathcal{D}_1$ with features from another dataset $\mathcal{D}_2$. We assume the owner of $\mathcal{D}_2$ may only be willing to share a privatised version of their dataset. For example, two organisations with mutual interests, such as the tax authorities and a private bank, or a fitness tracking company and a hospital, may want to join datasets to improve the performance of their algorithms. Similarly, it may be against regulations for multinational organisations to share and join non-privatised client data between departments in different regions, but permitted when the shared data satisfies LDP.

Table 4.6: Accuracy of classifiers trained on the join of clean and ϵ-LDP features of the Lending Club dataset. Each row shows the ϵ-LDP guarantee for the collected training set. The baseline refers to the accuracy when classifying clean features only.

	Privacy Level	VLM	Duchi	Laplace
Lending Club	$\epsilon_{\text{LDP}} = 10$	**61.2±0.8**	56.5±0.2	56.8±0.5
	$\epsilon_{\text{LDP}} = 8$	**60.5±0.4**	56.4±0.3	56.7±0.6
	$\epsilon_{\text{LDP}} = 6$	**59.4±0.4**	56.2±0.3	56.7±0.2
	$\epsilon_{\text{LDP}} = 4$	**57.9±0.7**	56.0±0.2	56.0±0.6
	$\epsilon_{\text{LDP}} = 2$	**56.5±0.1**	55.9±0.2	55.8±0.3
	$\epsilon_{\text{LDP}} = 1$	**56.1±0.1**	55.6±0.6	55.7±0.1
	No Privacy	64.8±0.2	65.8±0.6	65.8±0.6
	Baseline	56.1±0.5	56.1±0.5	56.1±0.5

We run this experiment on Lending Club. We split the datasets such that both $\mathcal{D}_1$ and $\mathcal{D}_2$ contain all rows, but $\mathcal{D}_1$ contains a subset of (clean) features, along with the clean label, and $\mathcal{D}_2$ contains the remaining features (to be privatised), as described in Appendix A.2.

We follow a privatisation procedure similar to that of the previous sections, with the distinction that the mechanism should be both trained on $\mathcal{D}_2$, and used to privatise $\mathcal{D}_2$. For the classification problem, instead of Equation 4.11 or 4.12, we optimise $\log p_\psi(y_1|x_1, \tilde{z}_2)$ where $(x_1, y_1) \in \mathcal{D}_1$ and $\tilde{z}_2$ is the privatised representation of some features $x_2 \in \mathcal{D}_2$. We have access to all raw data needed for validation, eliminating the need to conduct a private grid as in previous experiments. Unlike the previous experiments where we trained classifiers to act on clean data at inference time, here we train the classifier on a combination of both clean and privatised features, and classify this same combination of clean and privatised features at inference time.

Results are shown in Table 4.6. The baseline of 56.1% is the classification accuracy when using features from $\mathcal{D}_1$ only. Meanwhile, classifying on all (clean) features gives a 65.8% accuracy. Neither benchmark achieves more than a 0.4 percentage point accuracy increase over the baseline, whereas the VLM achieves a significant improvement for local $\epsilon \in [4, 10]$.

We note that, unlike in previous experiments, we do not require auxiliary data to

train the VLM here. Given a dataset $\mathcal{D}$ that we want to privatise and share, we can both train a mechanism using $\mathcal{D}$, and use that mechanism to privatise $\mathcal{D}$. This can be viewed as a specific case of the more general problem of private data publishing. However, unlike CDP synthetic data generation approaches [Acs et al., 2019, Takagi et al., 2021, Triastcyn and Faltings, 2019, Xie et al., 2018], each datapoint refers to a specific individual.

4.4 Classifying Private Datapoints

In Sections 4.3.1 and 4.3.2, we investigated the use of LDP data to train algorithms that classify clean datapoints at inference time. In some scenarios however, we may want to train algorithms that act directly on LDP datapoints at inference time. Most notably, in the data collection framework, the organisation may want to do inference on individuals whose data they have privately collected.

However it is clear from Definition 2.3.5 that privatising a data point under LDP will cause a considerable amount of information loss, limiting classification accuracy.

To quantify this information loss, it would be useful to calculate the maximum achievable accuracy of a K-class classifier $\mathcal{C}$ acting on ϵ-LDP datapoints privatised with the local Laplace mechanism. In Section 4.4.1, we discuss an upper bound for the maximum achievable accuracy, before introducing a tractable approximation to this maximum in Section 4.4.2.

4.4.1 General Upper Bound on Classification Accuracy

Recall we have some function $f(\cdot) : \mathcal{X} \to \mathcal{T}$ that maps data x to representations $z = f(x)$ inside a d-dimensional taxicab sphere $\mathcal{T}$ of diameter Δf. We pass x through a randomised algorithm $\mathcal{M} = f(\cdot) + (s_1, \ldots, s_d)$ to induce LDP, where $s_i \sim \text{Laplace}(0, b)$ and $b = \Delta f/\epsilon$. We denote the LDP point $\tilde{z}$.

We make the simplifying assumption that we have equal class balance, which is the case for all experiments in this work.

The classifier $\mathcal{C}$ will partition $\mathbb{R}^d$ into $\{S^{(1)}, \ldots, S^{(K)}\}$ such that $\tilde{z} \in S^{(k)}$ will be

classified into class k. The accuracy of $\mathcal{C}$ is then given by

$$A = \mathbb{E}_{(x,y)\sim p(x,y),\tilde{z}\sim\mathcal{M}(\tilde{z}|x)}\left[\mathbb{I}\left(\tilde{z}\in S^{(y)}\right)\right] \tag{4.13}$$

$$= \mathbb{E}_{(x,y)\sim p(x,y)}\left[\int_{\tilde{z}\in\mathbb{R}^d}\mathbb{I}\left(\tilde{z}\in S^{(y)}\right)\frac{1}{(2b)^d}e^{-\frac{||f(x)-\tilde{z}||_1}{b}}d\tilde{z}\right] \tag{4.14}$$

$$= \mathbb{E}_{(x,y)\sim p(x,y)}\left[\int_{\tilde{z}\in S^{(y)}}\frac{1}{(2b)^d}e^{-\frac{||f(x)-\tilde{z}||_1}{b}}d\tilde{z}\right] \tag{4.15}$$

where $p(x, y)$ is the true underlying data distribution. We observe that it is always possible to achieve a greater (or equal) classification accuracy if $f(\cdot)$ maps all data from a given class k to a single point on the taxicab sphere, rather than some region $\Gamma^k \subseteq \mathcal{T}$ containing more than one point.

To justify this, suppose $f(\cdot)$ maps all points from class k to some region Γ^k. The classifier defines a decision region $S^{(k)}$ such that any point inside $S^{(k)}$ gets classified as class k. There exists (at least one) point $c^{(k)} \in \Gamma^{(k)}$ such that $\forall g^{(k)} \in \Gamma^{(k)}$:

$$\int_{\tilde{z}\in S^{(k)}}\frac{1}{(2b)^d}e^{-\frac{||c^{(k)}-\tilde{z}||_1}{b}}d\tilde{z} \geq \int_{\tilde{z}\in S^{(k)}}\frac{1}{(2b)^d}e^{-\frac{||g^{k}-\tilde{z}||_1}{b}}d\tilde{z} \tag{4.16}$$

This says that $c^{(k)} \in \Gamma^{(k)}$ is the point such that the Laplace$(c^{(k)}, b)$ distribution contains more probability mass inside our decision region $S^{(k)}$ than any other distribution of the form Laplace$(g^{(k)}, b)$ with $g^{(k)} \in \Gamma^{(k)}$. In other words, $c^{(k)}$ is the representation inside $\Gamma^{(k)}$ most likely to still be classified as class k after privatisation. So if we modify $f(\cdot)$ such that all points from class k are mapped to a single representation $c^{(k)} \in \Gamma^{(k)}$, we will achieve higher (or equal) accuracy than with the original $f(\cdot)$.

In light of this, we assume $f(\cdot)$ maps all data from class k to a single representation $c^{(k)}$ and write Equation 4.15 as

$$A = \mathbb{E}_{y\sim p(y)}\left[\int_x\int_{\tilde{z}\in S^{(y)}}\frac{1}{(2b)^d}e^{-\frac{||f(x)-\tilde{z}||_1}{b}}d\tilde{z}\;p(x|y)dx\right] \tag{4.17}$$

$$= \mathbb{E}_{y\sim p(y)}\left[\int_{\tilde{z}\in S^{(y)}}\frac{1}{(2b)^d}e^{-\frac{||c^{(y)}-\tilde{z}||_1}{b}}d\tilde{z}\right] \tag{4.18}$$

$$= \frac{1}{K}\sum_{y=1}^{K}\int_{\tilde{z}\in S^{(y)}}\frac{1}{(2b)^d}e^{-\frac{||c^{(y)}-\tilde{z}||_1}{b}}d\tilde{z} \tag{4.19}$$

where the last inequality follows from the assumption of equal class balance, and the

integral inside the sum

$$A^{(y)} := \int_{\tilde{z} \in S^{(y)}} \frac{1}{(2b)^d} e^{-\frac{||c^{(y)} - \tilde{z}||_1}{b}} d\tilde{z} \tag{4.20}$$

represents the probability of correctly classifying a noised representation from class y, and is equal to the total probability mass of Laplace$(c^{(y)}, b\mathbb{I})$ inside $S^{(y)}$.

We know that the A will be maximised when the decision boundaries are such that a noised representation $\tilde{z}$ is classified as coming from class y if $c^{(y)}$ is the closest of the K representations $\{c^{(1)}, \ldots, c^{(K)}\}$ in ℓ_1-distance. This is because the probability mass at $\tilde{z}$ will be highest under a distribution with mean closest to $\tilde{z}$, and so $\tilde{z}$ will contribute more mass to $A^{(y)}$ than it would any other $A^{(\hat{y})}$ for $\hat{y} \neq y$. Thus $A = \frac{1}{K} \sum_{y=1}^{K} A^{(y)}$ will be higher.

In ℓ_1 geometry there exist subsets of $\mathbb{R}^d$ where all points in the subset lie equidistant from multiple class representations, as shown in Figure 4.3a. In this scenario, the choice of decision boundary through such subsets will not affect accuracy: assigning a region to class y rather than some equidistant class $\tilde{y} \neq y$ will increase $A^{(y)}$ and reduce $A^{(\tilde{y})}$ by the same amount, leaving A unchanged. We can therefore arbitrarily assign noised representations on such hyperplanes to any of the closest (in ℓ_1-norm) classes – a simple approach is to assign them to the class representation closest in L2-norm, as shown in Figure 4.3b.

In order to find the maximum achievable accuracy, we must find the representations $\{c^{(1)}, \ldots, c^{(K)}\}$ within $\mathcal{T}$ that maximise Equation 4.19, where our decision regions $S^{(y)}$ are defined as described above.

We note that $S^{(y)}$ correspond to Voronoi cells with generators $c^{(y)}$. Thus our problem is equivalent to finding the optimal positions of generators such that the probability mass inside each cell (under the distribution centred at that cell's generator) is maximised. Finding the optimal generator locations is an active field of research spanning fields such as computational geometry [Bhattacharya, 2010] and operations research [Riol et al., 2011]. Studies of probability density in Voronoi cells have appeared in fields such as Astrophysics [Jamieson and Loverde, 2021]. However, our task of finding the locations of the K optimal generators inside the d-dimensional

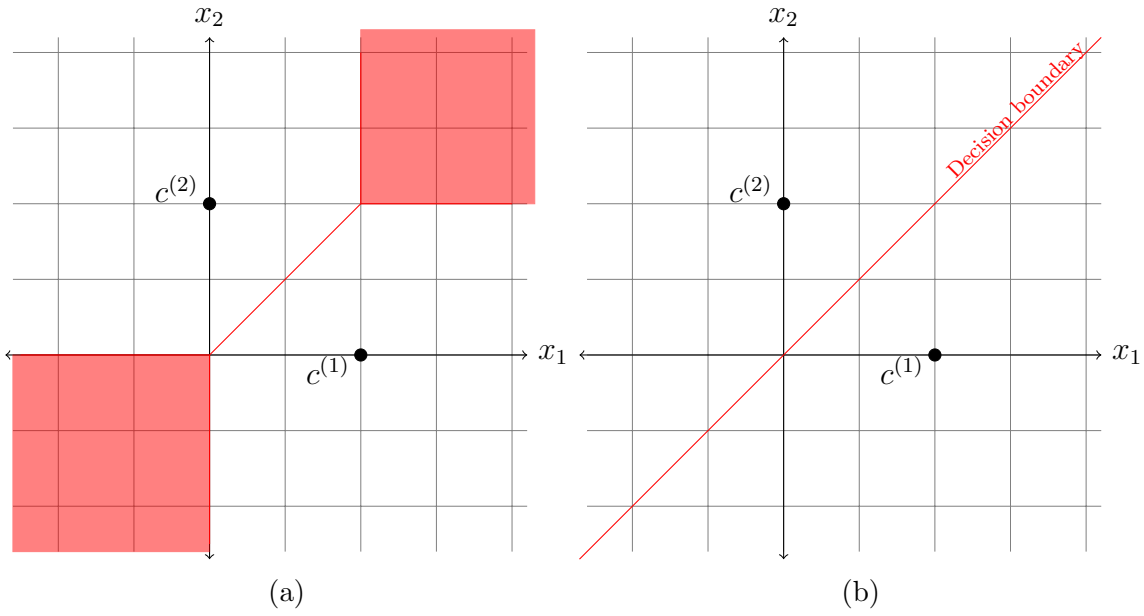


Figure 4.3: (a) Red shaded areas and lines represent the regions of $\mathbb{R}^2$ in which all points are equal L1 distance from $c^{(1)}$ and $c^{(2)}$. (b) The red line represents a decision boundary that separates $c^{(1)}$ and $c^{(2)}$ equally in L1 distance. Regions in which points are equidistant from representations $c^{(1)}$ and $c^{(2)}$ are divided based on the closest representation in L2 distance.

taxicab sphere is challenging. We leave a detailed analysis to future work and instead study a simplified setting which we hypothesise to be a good approximation to the optimal solution, for the setting defined in our experiments.

4.4.2 Simplified Setting

It seems reasonable to assume that when the taxicab sphere $\mathcal{T}$ has at least as many vertices as there are classes (i.e. $d \geq K/2$, as in all experiments in this work), the representations $\{c^{(1)}, \ldots, c^{(K)}\}$ that lead to the highest accuracy will lie on the vertices of $\mathcal{T}$. Furthermore, when $d > K/2$, we hypothesise it is favourable to place representations opposite one another (i.e. $c^{(i)} \cdot c^{(j)} = -\Delta f^2/4$)) rather than on different axes (i.e. $c^{(i)} \cdot c^{(j)} = 0$), where possible.

Supporting this assumption, when $d = K = 2$ it is straightforward to show that the accuracy is highest when $c^{(1)}$ and $c^{(2)}$ lie on opposite vertices rather than elsewhere on the boundary of $\mathcal{T}$. Furthermore, we ran numerical simulations of this problem which similarly concluded that for $d \in \{2, 3\}$ and $K \in \{2, \ldots, 6\}$, the

optimal setting is to place the representations on vertices of the sphere (and indeed opposite vertices when $d > K/2$). We found higher dimensional problems to be too computationally expensive to simulate numerically.

In light of this, we construct a setting that places representations on (opposite) vertices of $\mathcal{T}$, which we expect to be a good approximation of the true maximum achievable accuracy when $d \geq K/2$. We define our representations $\{c^{(1)}, \ldots, c^{(K)}\}$ as

$$c^{(k)} = \begin{cases} \frac{1}{2}\Delta f \cdot e_{\frac{k}{2}}, & \text{for } k \text{ odd} \\ -\frac{1}{2}\Delta f \cdot e_{\frac{k+1}{2}}, & \text{for } k \text{ even} \end{cases} \tag{4.21}$$

where e_i is the i^{th} standard basis vector. This places representations on opposite vertices of the taxicab sphere, for each axis in turn, until all representations have been assigned. Corresponding decision regions are defined as in Section 4.4.1. Figure 4.4 shows these representations and corresponding decision boundaries for $d = 2$ and $K = 4$.

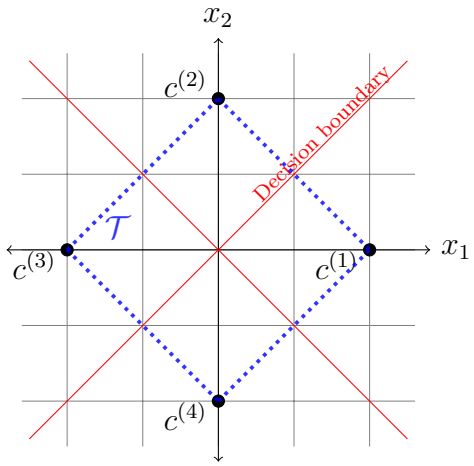


Figure 4.4: The decision boundary for a classifier that equally separates (in ℓ_1-distance) vertices $c^{(i)}$ for $i \in \{1, 2, 3, 4\}$ in 2-dimensional space. The blue region denotes the taxicab sphere $\mathcal{T}$.

Given these representations, and the inferred decision regions, we can calculate the accuracy of the optimal classifier given by Equation 4.19. We first consider the case of $K = 2d$, meaning we have the same number of vertices as classes. To ease notation we assume without loss of generality that $\Delta f = 2$, and so $c^{(1)} = (1, 0, \ldots, 0)$.

We also denote $\tilde{z} = (\tilde{z}_1, \ldots, \tilde{z}_d)$.

First, by symmetry, it is clear that $A^{(1)} = A^{(2)} = \ldots = A^{(K)}$, and so $A = A^{(1)}$. The decision boundary $S^{(1)}$ is defined as

$$S^{(1)} = \{(\tilde{z}_1, \ldots \tilde{z}_d) : \ \tilde{z}_1 > 0, \ \text{ and } \ \tilde{z}_i < |\tilde{z}_1|, \ \forall i \in \{2, \ldots, K/2\}\} \tag{4.22}$$

We can then calculate the accuracy A as follows:

$$\begin{aligned}
A &= \int_{\tilde{z} \in S^{(1)}} \frac{1}{(2b)^d} e^{-\frac{||c^{(1)} - \tilde{z}||_1}{b}} d\tilde{z} && (4.23)\\
&= \int_{\substack{\tilde{z}_1 > 0, \\ \tilde{z}_i < |\tilde{z}_1|, \forall i \neq 1}} \frac{1}{(2b)^d} e^{-\frac{||c^{(1)} - \tilde{z}||_1}{b}} d\tilde{z}_{1:d} && (4.24)\\
&= \int_0^\infty \frac{1}{2b} e^{-\frac{|1-\tilde{z}_1|}{b}} \left(\prod_{i=2}^{d} \int_{-\tilde{z}_1}^{\tilde{z}_1} \frac{1}{2b} e^{-\frac{|\tilde{z}_i|}{b}} d\tilde{z}_i \right) d\tilde{z}_1 && (4.25)\\
&= \int_0^\infty \frac{1}{2b} e^{-\frac{|1-\tilde{z}_1|}{b}} \left(1 - e^{-\tilde{z}_1/b} \right)^{d-1} d\tilde{z}_1 && (4.26)\\
&= \int_0^\infty \frac{1}{2b} e^{-\frac{|1-\tilde{z}_1|}{b}} \sum_{j=0}^{d-1} \binom{d-1}{j} (-1)^j e^{-\frac{j\tilde{z}_1}{b}} d\tilde{z}_1 && (4.27)\\
&= \sum_{j=0}^{d-1} \binom{d-1}{j} (-1)^j \int_0^\infty \frac{1}{2b} e^{-\frac{|1-\tilde{z}_1|}{b}} e^{-\frac{j\tilde{z}_1}{b}} d\tilde{z}_1 && (4.28)\\
&= \sum_{j=0}^{d-1} \binom{d-1}{j} (-1)^j \left[\int_0^1 \frac{1}{2b} e^{-\frac{\tilde{z}_1(j-1)+1}{b}} d\tilde{z}_1 + \int_1^\infty \frac{1}{2b} e^{-\frac{\tilde{z}_1(j+1)-1}{b}} d\tilde{z}_1 \right] && (4.29)\\
&= \frac{1-d}{2b} \left(1 + \frac{b}{2} \right) e^{-1/b} + \sum_{j=0, j\neq 1}^{d-1} \binom{d-1}{j} (-1)^j \left[\frac{e^{-j/b}}{1-j^2} - \frac{e^{-1/b}}{2(1-j)} \right] && (4.30)\\
&= (1-d) \frac{\epsilon+1}{4} e^{-\epsilon/2} + \sum_{j=0, j\neq 1}^{d-1} \binom{d-1}{j} (-1)^j \left[\frac{e^{-j\epsilon/2}}{1-j^2} - \frac{e^{-\epsilon/2}}{2(1-j)} \right] && (4.31)\\
&= \sum_{j=0, j\neq 1}^{K/2-1} \left(\binom{K/2-1}{j} \frac{(-1)^j}{1-j} \left[\frac{e^{-j\epsilon/2}}{1+j} - \frac{e^{-\epsilon/2}}{2} \right] \right) - \frac{\epsilon+1}{8} (K-2) e^{-\epsilon/2} && (4.32)
\end{aligned}$$

where in the penultimate step we used the fact that for ϵ-LDP we have $b = 2/\epsilon$, and in the final equality we substitute $d = K/2$.

We now consider the case where $K \leq 2d$ and K is even. In this case, the decision

Table 4.7: Private Accuracy of classifiers trained on ϵ_{train}-LDP (image, label) tuples collected using different LDP mechanisms. ϵ_{test} refers to the LDP guarantee of the images classified at inference time. Error bars represent ±1 standard deviation from the mean over 3 trials.

	Privacy Level	VLM	PrivUnit	Duchi	Laplace	Upper Bound
MNIST	$\epsilon_{\text{TRAIN}} = 10$, $\epsilon_{\text{TEST}} = 7$	**42.3±0.5**	9.6±0.3	10.1±0.4	10.1±0.3	80.7
	$\epsilon_{\text{TRAIN}} = 8$, $\epsilon_{\text{TEST}} = 5.6$	**37.7±0.5**	10.0±1.3	10.1±0.3	10.7±0.8	69.3
	$\epsilon_{\text{TRAIN}} = 6$, $\epsilon_{\text{TEST}} = 4.2$	**31.7±1.2**	10.3±0.4	10.4±0.1	11.0±0.5	53.8
	$\epsilon_{\text{TRAIN}} = 4$, $\epsilon_{\text{TEST}} = 2.8$	**20.1±0.6**	11.0±0.7	10.1±0.9	9.8±0.9	36.0
	$\epsilon_{\text{TRAIN}} = 2$, $\epsilon_{\text{TEST}} = 1.4$	**10.5±0.6**	10.3±0.2	9.7±0.9	9.9±0.3	20.0
	$\epsilon_{\text{TRAIN}} = 1$, $\epsilon_{\text{TEST}} = 0.7$	10.2±0.5	**10.8±1.2**	10.2±0.7	10.6±1.3	14.2
	No Privacy	94.9±0.2	96.0±0.4	96.0±0.4	96.0±0.4	100.0

boundary $S^{(1)}$ is defined as

$$S^{(1)} = \{(\tilde{z}_1, \dots \tilde{z}_d) : \tilde{z}_1 > 0, \\ \tilde{z}_i < |\tilde{z}_1|,\ \forall i \in \{2, \dots, K/2\}, \\ \tilde{z}_j \in (-\infty, \infty),\ \forall j \in \{K/2+1, \dots, d\}\} \tag{4.33}$$

where the unbounded dimensions integrate to 1, leaving accuracy unchanged. Equation 4.32 therefore defines the maximum achievable accuracy in this simplified setting, and a proxy for the maximum achievable accuracy of the private classifiers studied in this work. We omit the case when K is odd, since K is even in all experiments.

4.4.3 Experimental Results

In Table 4.7, we show the accuracy of classifiers when applied to privatised datapoints at inference time. Unlike in previous experiments, two epsilon values are given. The first, ϵ_{train}, indicates the privacy guarantee of the classifier training set made up of (data, label) pairs collected with the VLM and flip mechanism respectively; in these experiments we assign $\lambda = 70\%$ of the privacy budget to the datapoint and the remaining 30% to the label (see Section 4.2.2 for details). The second, ϵ_{test}, indicates the privacy value of the privatised test points on which we test our classifier (since

no label is collected, these values are 70% of ϵ_{train}. We compare our empirical results to the upper bound in Equation 4.32.

Running experiments on MNIST, we see a considerable drop in performance when classifying privatised datapoints, compared with results from Section 4.3.1. While we are clearly not achieving the accuracy from Equation 4.32 (denoted 'Upper Bound' in the table), we note that our method aims to build a downstream-task-agnostic, privatised representation of the data. Thus the representation must contain more information than just the class label. Meanwhile, the upper bound is derived from the extreme setting in which the representation encodes only class information, and would be unable to solve any other downstream task.

4.5 Conclusion

In this chapter, we have taken an important first step in the privatisation of high-dimensional data under LDP, clearly demonstrating that dimensionality-reduction is a compelling approach to overcoming the significant hurdles of privatisation in high-dimensions. Our work represents the first use of latent variable modelling for LDP data collection, and can be easily adapted to any data type for which latent variable modelling is possible. We demonstrate a range of applications, spanning important issues such as medical diagnosis, financial crime detection, and customer experience improvement, significantly outperforming existing baselines throughout.

We note however, that while the data studied in this chapter is considered high-dimensional in the context of LDP, it is no longer considered high-dimensional in the broader machine learning context. For maximum impact, we must therefore consider higher-dimensional problems. We explore this in next chapter.

Chapter 5

High-Dimensional Representation Learning under LDP

The work presented in this chapter was published in Mansbridge et al. [2022].

In the previous chapter, we developed a generative latent variable model to privatise high-dimensional data through the addition of carefully calibrated noise to the constrained latent representations. By privatising in low-dimensional latent space we circumvent many of the problems associated with inducing LDP in high-dimensions. Furthermore, we introduced a novel, 'denoising' approach for classifier training that models the relationship between (unobserved) clean datapoints and labels, given access only to privatised datapoints and labels at training time.

In this chapter, our goal is to privatise even higher-dimensional data by scaling up the ideas introduced in Chapter 4. As before, our evaluative task is to use the privatised data to train downstream machine learning models, but these models now solve tasks that are considered difficult even in a non-private setting. To achieve this, our mechanisms are based on current state-of-the-art representation learning approaches, demonstrating that the ideas introduced in the previous chapter can be applied not only to generative latent variable models, but more broadly across the field of representation learning.

5.1 Introduction

Advancements in machine learning research, stemming in part from more powerful compute resources, easy access to vast quantities of data, and significant developments in algorithm design have meant that both the complexity of data on which we do inference, and the breadth of applications for such models, have increased dramatically. As such, while the tabular data and small images discussed in Chapter 4 are considered high-dimensional in the context of mechanisms for local differential privacy (LDP), they are no longer considered high-dimensional in the broader context of (non-private) machine learning research. In line with this, there exists a growing need for mechanisms which facilitate the collection of even higher-dimensional user data under LDP – an area of research that is severely limited at present.

Naively, one could use a VLM to privatise this complex, high-dimensional data, perhaps adopting more powerful architectures in the encoder and decoder networks. However, as discussed in Chapter 3, this may not necessarily lead to meaningful latent representations. Instead, we develop LDP mechanisms through the adoption of state-of-the-art techniques from the broader field of representation learning, which has witnessed significant progress on high-dimensional tasks in recent years.

The way in which we represent information can heavily influence the difficulty of tasks. If you were to ask someone what they were doing yesterday at midday, they would likely be able to answer easily. However, if you replace "yesterday at midday" with the date in UNIX time (that is, the number of seconds elapsed since 00:00:00 UTC on 1 January 1970), the question becomes more difficult for a human to understand. On the other hand, if you were to ask somebody to calculate their age in seconds, the somewhat cumbersome calculation is reduced to a simple subtraction if you provide the respondent with both the current time and their time of birth in UNIX time.

Data representations affect the difficulty of information processing for computers, just as they do for humans. The goal of representation learning is to take complex, real-world data and map it to some expressive, meaningful representation that is useful for downstream tasks. Whilst domain specific knowledge can be used to construct

such representations, a primary goal is to *learn* these explanatory factors in the data. Learning such representations has significant implications across machine learning. For example, it can improve data efficiency, increase the robustness of algorithms, and improve generalisation. The exact nature of this mapping, and what we define as 'useful' can vary considerably with application. One may want to learn disentangled or interpretable representations [Locatello et al., 2019], representations suitable for specific tasks [Ballé et al., 2018, Frye et al., 2021], or indeed representations that generalise well, making them suitable for downstream learning [Devlin et al., 2019, Chen et al., 2020]. In this chapter, we are interested in learning representations that are both useful for downstream learning, and robust to the noise requirements of LDP.

Representations can be learnt in a supervised or unsupervised manner. Supervised representation learning can be as simple as training a neural network classifier; each layer of the network can be seen as learning some sort of representation of the input data. If a softmax activation is applied to the final hidden layer of a network, one might hope this layer learns some linearly separable representation, whilst previous layers may exhibit other valuable properties. Machine learning scientists frequently make use of representations obtained from powerful pre-trained convolutional networks like ResNets [He et al., 2016] or Inception networks [Szegedy et al., 2016] to solve a range of computer vision tasks.

In recent years, both the number of parameters in state-of-the-art networks, and the quantity of training data required to train these networks has increased significantly. Collecting labelled data can be expensive, time consuming, and when a third party is hired to label the data, a potential privacy risk. Unsupervised approaches have been utilised extensively for learning in recent years to eliminate this need for labelled data. These models can be broadly categorised as either generative, or non-generative models.

Latent variable models (LVMs), such as the variational autoencoder (VAE), constitute unsupervised, generative representation learning models. While VAEs have shown compelling results on a broad range of data types, for very high-dimensional data they typically rely on deep hierarchies of latents to generate realistic data

samples, with the total number of latent dimensions often exceeding the original data dimension [Maaløe et al., 2019, Child, 2021]. Despite these models achieving state-of-the-art performance in terms of log likelihood, it is not immediately obvious how one would efficiently privatise such latent hierarchies, whether these latents provide meaningful representations, or indeed whether they would be useful as tools for downstream model training.

One compelling feature of VAE-based LDP mechanisms is that they facilitate the private collection of data on original feature level, by passing the privatised data representation through the decoder network, as discussed in Section 4.3.1.1. However, the joint training of both a generative distribution $p(x|z)$ and inference distribution $q(z|x)$ presents optimisation challenges, since the two networks are highly entangled. In the extreme (and not uncommon) case, the model may exhibit 'posterior collapse', whereby no information is encoded, as discussed in Section 3.2. In many applications of private data collection, it is sufficient to collect privatised data on representation level. In this scenario, the generative distribution is not used at privatisation time, rendering the added complexity, optimisation challenges, and computational expense of training it unnecessary.

Non-generative models do not generally learn the mapping from representation space back to latent space and so circumvent many of these issues. Self-supervised learning has led to some of the most significant developments in representation learning in recent years, providing a framework for training models in a 'supervised' manner with unlabelled data by constructing synthesised labels. This has been used extensively to train large natural language processing models with impressive empirical performance. Devlin et al. [2019] learn powerful representations by randomly masking words in sentences, and using them as labels for missing word prediction. They also train the model on next sentence prediction, feeding in contiguous sentences from large corpora as positive examples and non-contiguous sentences as negative examples. Variants on these techniques have also been used to learn sentence-level representations [Yang et al., 2021]. In computer vision, state-of-the-art image representations have also been achieved with self-supervised methods. Chen et al. [2020] and He et al. [2020] randomly augment each image in the batch twice, labelling

augmentations of the same image as positive pairs, and augmentations of different images as negative pairs. The model is then trained to minimise cosine similarity between representations of positive pairs and maximise cosine similarity between negative pairs. In doing so, image representations are encouraged to be invariant to transformations such as flipping or blurring, whilst dissimilar images should have dissimilar representations.

In Section 5.2, we build on ideas described in Chapter 4 and introduce a systematic and straightforward procedure for adapting existing representation learning algorithms, such that the learnt representations are robust to LDP-inducing noise. We then use this approach to develop an LDP mechanism for colour image data in Section 5.2.2, adapting a contrastive representation learning model. Contrastive models [van den Oord et al., 2018, Chen et al., 2020, He et al., 2020] are naturally suited here, having achieved state-of-the-art performance in representation learning for high-dimensional images. In Section 5.3, we test this mechanism empirically on the CIFAR-10 colour image dataset [Krizhevsky, 2009], achieving significant performance gains over current state-of-the-art mechanisms. As with the VLM, we measure the utility of our privatised representations by using them to train downstream machine learning models, which is a common goal for many data collectors. We dub this mechanism the contrastive Laplace mechanism (CLM).

5.2 Representation Learning Laplace Mechanism

As in Chapter 4, we assume access to an unlabelled, auxiliary dataset from a similar distribution to the data we hope to collect. We use this auxiliary dataset to train the LDP mechanism used to collect new data.

We outline a straightforward procedure for training an LDP mechanism through the modification of an arbitrary representation learning model. We assume this model learns a representation of a datapoint x:

$$z = f_\phi(x) \tag{5.1}$$

where f_ϕ is a function that maps from data space to (non-private) representation space, and is parameterised by a neural network with parameters ϕ.

Our LDP mechanism maps data through the modified $f_\phi(\cdot)$ before inducing privacy on the inferred representation. The performance of this mechanism is heavily dependent on the quality of the learnt mapping $f_\phi(\cdot)$, which must satisfy three crucial properties:

1. The output of $f_\phi(\cdot)$ must be constrained such that the addition of calibrated noise guarantees LDP.

2. The inferred representations must contain as much information as possible for downstream model learning.

3. We need the representations to retain as much of this information as possible *after* privatisation.

As in Chapter 4, the first property can be easily achieved by applying a suitable activation function $\nu(\cdot)$ to the final layer of the neural network i.e. $f_\phi(\cdot) = \nu(h_\phi(\cdot))$. With the VLM, we were restricted to LDP mechanisms based on reparameterisable noise distributions but, in general, we have greater flexibility over our choice of mechanism.

Several LDP mechanisms, like the local Gaussian mechanism [Wang et al., 2021] and the local Laplace mechanism, require finite ℓ_p-sensitivity. One can restrict the sensitivity to $\Delta f_\phi = 2l$ (for some $l > 0$) with the following activation:

$$\nu_p(h) = h * \min\{1, l/||h||_p\} \tag{5.2}$$

Different activations can be used for mechanisms with different requirements. In this chapter, the goal is to demonstrate the efficacy of representation learning for LDP, rather than to compare existing mechanisms in representation space. Thus we focus on the Laplace mechanism as before, inducing ϵ-LDP with the mechanism:

$$\mathcal{M}\left(x, f_\phi(x), \epsilon\right) = f_\phi(x) + (s_1, \ldots, s_k) \tag{5.3}$$

where $f_\phi(x) := \nu_1(h_\phi(x))$ and $s_i \sim \text{Laplace}(0, 2l/\epsilon_x)$.

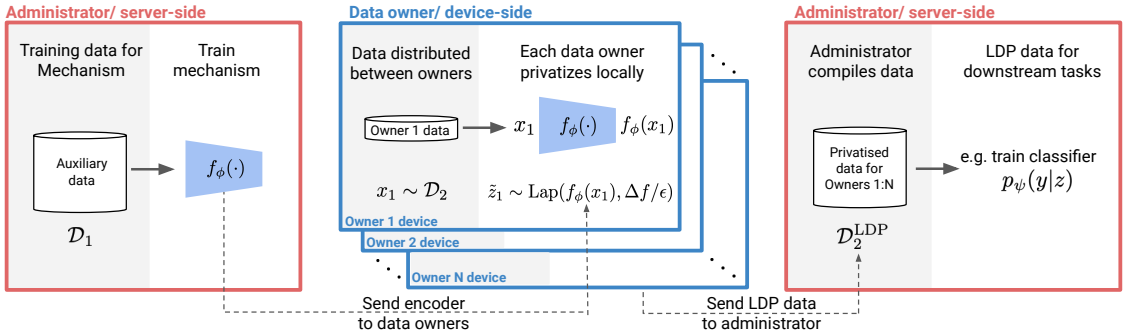


Figure 5.1: Schematic diagram of mechanism training (left), local data privatisation (centre) and collection (right), as outlined in Section 5.2. Red boxes indicate operations performed on the administrator/ data collector's infrastructure and blue boxes indicate operations performed locally by the data owner. Crucially, unprivatised data never leaves the data owner's device.

The second property can be achieved by our choice of representation learning algorithm. Downstream model learning is a key task within non-LDP representation learning, and so by adapting a non-LDP model that achieves state-of-the-art performance on downstream learning for the datatype at hand, we can be confident that our mechanism will learn powerful representations. For example, in Section 5.2.2, we privatise high-dimensional colour images, using a modified version of SimCLR [Chen et al., 2018], which achieved accuracy on downstream tasks comparable to that of direct supervised learning in the non-private setting.

To satisfy the final property, we must ensure the learnt representations are maximally-robust to LDP-inducing noise. We found a straightforward and effective approach to be the addition of noise to the representations during mechanism training. This additive noise should be from the same distribution as in the LDP mechanism, though not necessarily as high variance. Suppose the original representation learning algorithm defines some loss function

$$\mathcal{L}^{\text{original}} \equiv \mathcal{L}\big(\{z_n\}_{n=1}^N\big), \qquad \text{where} \quad z_n = f_\phi(x_n) \tag{5.4}$$

over the training set $\{x_n\}_{n=1}^N$. Then to learn say, a Laplace mechanism, we instead optimise

$$\mathcal{L}^{\text{robust}} \equiv \mathcal{L}\big(\{\tilde{z}_n\}_{n=1}^N\big), \qquad \text{where} \quad \tilde{z}_n \sim \text{Laplace}(\nu_1(h_\phi(x_n)), b\mathbb{I}) \tag{5.5}$$

where b is treated as a hyperparameter found using a private grid search, as in Section 4.2.4. We note that other model-specific techniques for encouraging noise robustness could be used instead, as we discuss in Section 5.2.2.

At privatisation time, a sample $\tilde{z} \sim \text{Laplace}(\nu_1(h_\phi(x)), \frac{2l}{\epsilon_x}\mathbb{I})$ forms an ϵ_x-LDP representation of x.

Downstream model training Our goal is to use our collected data for supervised downstream model training, and so we also privatise and collect corresponding labels alongside the LDP representations. We use the same flip mechanism as in Section 4.2.2 to collect labels y under ϵ_y-LDP so that, by the composition theorem, our labelled privatised data tuples $(\tilde{z}, \tilde{y})$ satisfy ϵ-LDP, where $\epsilon = \epsilon_x + \epsilon_y$.

We then use the classifiers introduced in Section 4.2.3 to train downstream models. Figure 5.1 shows a schematic diagram of mechanism training, data collection, and downstream model training.

Using the approach outlined in Section 5.2, we introduce a mechanism designed specifically for the privatisation of high-dimensional colour image data. We adapt SimCLR [Chen et al., 2020] to learn powerful, LDP representations that are highly effective for downstream model training.

5.2.1 SimCLR

SimCLR is a contrastive learning model for learning representations of large colour images. To train the model, each image in the minibatch $\{x_b\}_{b=1}^B$ is augmented twice via randomly sampled transforms $t \sim \text{T}$. These transformations involve:

- Random cropping and resizing to 224x224,
- Random flipping with probability 0.5,
- Colour jitter applied with probability 0.8,
- Conversion to grayscale with probability 0.2.

These augmentations $\{\bar{x}_{b'}\}_{b'=1}^{2B}$ are mapped through a ResNet $f_\phi(\cdot)$ giving $2B$ representations $\{z_{b'}\}_{b'=1}^{2B}$, where for each positive pair (representations of augmen-

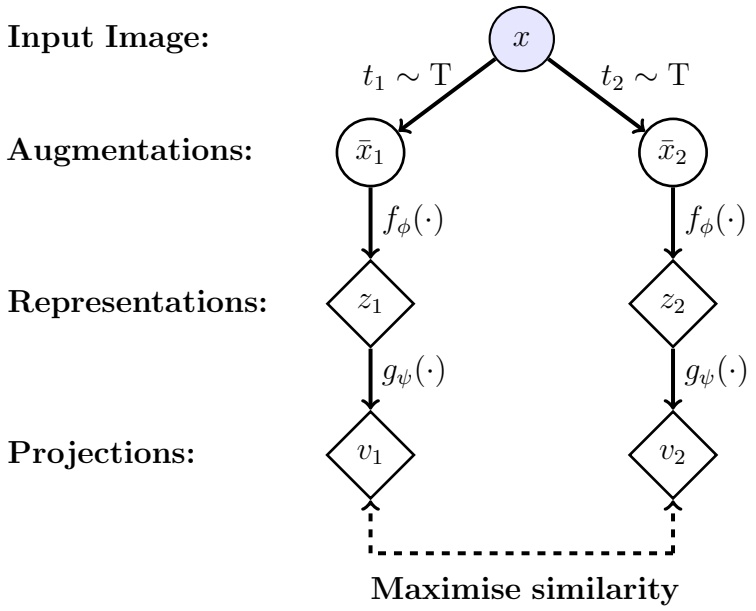


Figure 5.2: Graphical representation of the SimCLR model [Chen et al., 2020]. The blue shading indicates that the random variable is observed.

tations of the same image) we have $2(B-1)$ negative pairs (representations of augmentations of different images). The representations are passed through a MLP $g_\psi(\cdot)$, known as a projection head, to give $v_{b'} = g_\psi(z_{b'}) = W^{(2)}\sigma(W^{(1)}(z_{b'}))$. The model then encourages these projected representations to be similar for positive pairs and dissimilar for negative pairs. This is achieved through the maximisation of a softmax over cosine similarities $\mathrm{sim}(v, w) = v^T w/||v||\,||w||$ between positive pairs (i, j):

$$\ell_{ij} = -\log \frac{\exp\big(\mathrm{sim}(v_i, v_j)\big)}{\sum_{k=1,k\neq i}^{2B} \mathrm{sim}(v_i, v_k)} \tag{5.6}$$

A graphical representation of this training process is shown in Figure 5.2.

5.2.2 Contrastive Laplace Mechanism

We construct an LDP mechanism trained with contrastive learning, by adapting SimCLR as described in Section 5.2. We use a ResNet-18 model for $f_\phi(\cdot)$, followed by a final hidden layer of size 32, and the activation $\nu_1(\cdot)$ defined in Equation 5.2. We induce LDP by adding Laplace$(0, \frac{2l\mathbb{I}}{\epsilon_x})$ noise to the representations $z = f_\phi(x)$. In doing so, we obtain ϵ_x-LDP representations $\tilde{z}$. This privatisation procedure is given

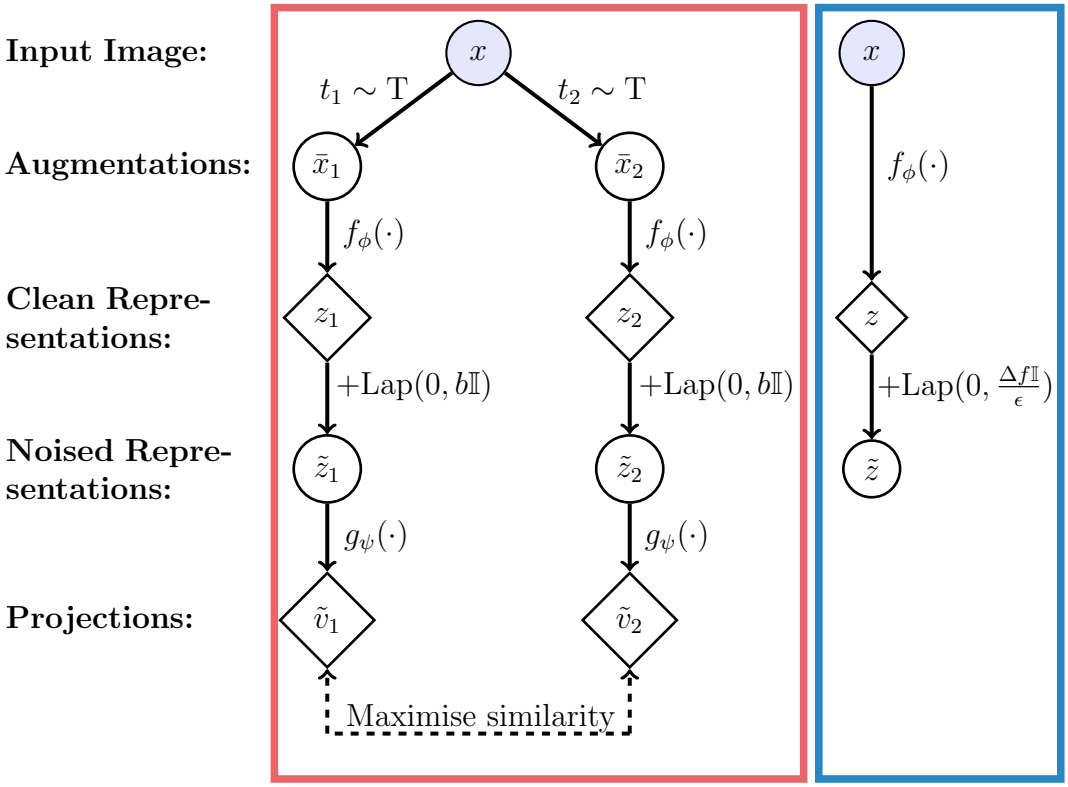


Figure 5.3: Graphical representation of the contrastive Laplace mechanism. The blue shading indicates that the random variable is observed. The red box shows the training procedure, performed on the administrator/ data collector's infrastructure; the blue box depicts the privatisation procedure, performed locally by the data owner.

in Figure 5.3 (right).

The training process is shown in Figure 5.3 (left). During training, we augment each image twice, as before, passing each noised augmentation representation through a MLP projection head as in SimCLR, to give

$$\tilde{v}_i = g_\psi(z_i + s_i) \equiv W^{(2)}\sigma(W^{(1)}(z_i + s_i)) \tag{5.7}$$

where $s_i \sim \mathrm{Laplace}(0, b\mathbb{I})$ and b is a model hyperparameter. We then maximise a softmax over cosine similarities between positive pairs (i, j):

$$\ell_{ij} = -\log \frac{\exp\big(\mathrm{sim}(\tilde{v}_i, \tilde{v}_j)\big)}{\sum_{k=1}^{2B} \mathrm{sim}(\tilde{v}_i, \tilde{v}_k)} \tag{5.8}$$

with respect to ϕ and ψ.

This approach assumes representations of images should be invariant to transformations like cropping, flipping or colour alterations, whilst representations of different images should be well separated (in terms of cosine similarity). This is well aligned with our goal of LDP representation learning: not only do we want representations that are well-suited to downstream model training, but we also want images that are different (for example, images from different classes) to be well separated in representation space so that they are robust to LDP-inducing noise.

Since robustness to Laplace noise equates to large separation in ℓ_1 distance, we also experimented with ℓ_1-based similarity metrics. Empirically however, we found a significant decrease in representation quality (in the absence of noise) compared with cosine similarity. The best trade-off between representation quality and noise-robustness was achieved using cosine similarity with additive Laplace noise in representation space during training, as with the VLM. We refer to this method as the contrastive Laplace mechanism (CLM).

We conduct a LDP grid search over mechanism hyperparameters as described in Section 4.2.4. After training the CLM, we use the privatised labelled ϵ-LDP representations $(\tilde{z}, \tilde{y})$, to train a classifier, using the approach introduced in Section 4.2.3. In doing so we are able to use classifier performance as a proxy for the utility retention in our LDP training set, as before.

5.2.2.1 VLM vs. CLM

We do not consider the CLM to be a replacement for the VLM. Rather, we recommend the data collector choose a mechanism appropriate for the data type and task at hand. For large images, the CLM is a natural choice of model. For smaller images or tabular data, where data dimension is smaller and the CLM's colour image augmentations are not applicable, the VLM is a more appropriate (and less computationally expensive) choice. Indeed, there exist a plethora of performant representation learning algorithms for a broad range of data modalities that one could adapt for privatisation. For example, large transformer models have recently shown success in representation learning for language data [Devlin et al., 2019, Yang et al., 2021], and so these are likely a good choice for learning LDP mechanisms for language data.

5.3 Applications and Experiments

We conduct experiments on the CIFAR-10 dataset [Krizhevsky, 2009] – a dataset containing colour images of size $32 \times 32 \times 3$. In Sections 5.3.1 and 5.3.2 we follow the experimental setup from the data collection and novel class classification experiments of Sections 4.3.1 and 4.3.2 respectively. In these experiments we train the CLM with the aim of using the collected, LDP data to train downstream models that act on clean (i.e. unprivatised) data at inference time. In Section 5.4, we follow the experimental setup from Section 4.4, training the CLM with the aim of training downstream models that act on LDP data at inference time.

Full experimental details can be found in Appendix B.1, and details of the private grid search over model hyperparameters found in Appendix B.3.

Benchmarks: We benchmark results against the Laplace mechanism, Duchi's mechanism [Duchi et al., 2018] and the PrivUnit$_2$/ScalarDP mechanism [Bhowmick et al., 2019], of which the latter two represent current state-of-the-art. Appendix B.2 covers implementation details for benchmark mechanisms. As in Chapter 4, we emphasise that classifiers trained with DP-SGD [Abadi et al., 2016] do not constitute meaningful benchmarks – we are interested in mechanisms for LDP data collection that maximise utility retention. Amongst other things, the collected LDP data can facilitate the training of classifiers, but the privacy of training set members is crucially never violated.

5.3.1 Data Collection

In this section, we follow the experimental setup of Section 4.3.1. We split the CIFAR-10 dataset such that 75% forms the auxiliary dataset $\mathcal{D}_1$ for training the CLM and the remaining 25% forms the dataset $\mathcal{D}_2$ that we privatise, collect, and use to train our classifier. We optimise Equation 4.11 when training a classifier on our LDP representations. Motivations and applications for this experiment are discussed in Section 4.3.1.

Table 5.1 outlines the performance of classifiers trained on LDP data collected

Table 5.1: Accuracy of classifiers trained on data collected using different LDP mechanisms. Each row shows the ϵ-LDP guarantee for the collected training set. Error bars represent ± 1 standard deviation from the mean over 3 trials.

	Privacy Level	CLM	PrivUnit	Duchi	Laplace
CIFAR-10	$\epsilon_{\text{LDP}} = 10$	**75.9±3.8**	18.9±1.1	12.8±1.1	10.2±0.4
	$\epsilon_{\text{LDP}} = 8$	**75.3±1.5**	13.2±2.9	11.1±0.7	9.9±0.4
	$\epsilon_{\text{LDP}} = 6$	**73.9±3.4**	11.0±2.0	10.6±0.7	9.9±0.1
	$\epsilon_{\text{LDP}} = 4$	**43.4±11.8**	9.6±0.4	10.9±0.9	9.6±0.5
	$\epsilon_{\text{LDP}} = 2$	**17.6±3.9**	10.1±0.1	10.4±0.8	10.0±0.1
	$\epsilon_{\text{LDP}} = 1$	**15.0±4.0**	9.5±0.5	9.9±0.1	9.8±0.1
	No Privacy	86.3±0.0	76.1±0.7	76.1±0.7	76.1±0.7

under different mechanisms, at various ϵ-LDP privacy guarantees. We clearly see that classifiers trained on data privatised with the CLM outperform those trained on data privatised with benchmark mechanisms at every ϵ value tested. Indeed, the CLM consistently achieves more than 50 percentage points higher than the benchmarks for $\epsilon \geq 6$, and achieves above random accuracy at all ϵ values. All benchmarks perform at, or close to, random accuracy for all $\epsilon \leq 8$. This indicates a far greater retention of utility when privatising with the CLM than with any of the benchmarks.

The 'no privacy' row demonstrates the advantages of using a representation learning approach: for the benchmarks, we simply train a classifier directly on the clean images, whilst for the CLM we train a classifier on learnt (non-LDP) representations. Since the LDP classifier training set $\mathcal{D}_2$ contains only 25% of the original CIFAR-10 training set (12,500 images), the benchmarks only achieve 76.1% accuracy, whilst the CLM approach, which only has to learn the simpler mapping from representation to label, achieves 86.3% accuracy. Indeed, even at $\epsilon = 10$, the CLM is still outperforming the non-private benchmark classifier.

5.3.2 Novel-Class Classification

In this section, we follow the experimental setup in Section 4.3.2, testing the performance of the CLM when the distribution of the auxiliary dataset $\mathcal{D}_1$ differs from the data $\mathcal{D}_2$ we wish to collect. Specifically, the downstream task is to predict member-

Table 5.2: Accuracy of classifiers for novel class classification, trained on data collected using different LDP mechanisms. Each row shows the ϵ-LDP guarantee for the collected training set. Error bars represent ± 1 standard deviation from the mean over 3 trials.

	Privacy Level	CLM	PrivUnit	Duchi	Laplace
CIFAR-10	$\epsilon_{\text{LDP}} = 10$	**89.3±0.5**	68.3±1.0	58.4±9.7	50.4±0.9
	$\epsilon_{\text{LDP}} = 8$	**89.5±0.5**	66.7±2.0	57.8±9.4	50.2±0.6
	$\epsilon_{\text{LDP}} = 6$	**89.7±0.6**	55.5±1.2	57.2±9.3	51.0±2.7
	$\epsilon_{\text{LDP}} = 4$	**89.4±0.4**	52.1±0.6	56.9±8.9	49.1±0.6
	$\epsilon_{\text{LDP}} = 2$	**87.2±1.4**	50.3±1.2	56.5±6.6	50.2±0.3
	$\epsilon_{\text{LDP}} = 1$	**82.8±2.1**	50.3±2.0	52.6±2.8	49.8±0.4
	No Privacy	94.3±0.1	92.2±0.4	92.2±0.4	92.2±0.4

ship of a class not present in $\mathcal{D}_1$. We split the CIFAR-10 dataset in the same way as in Section 4.3.2, such that $\mathcal{D}_1$ contains data from classes 0 to 8 and $\mathcal{D}_2$ contains an equal split of 9's and 'not 9's'. As before, we found classifier performance was best when trained by maximising $\log p(y|\tilde{z})$ directly, rather than with the objective that marginalises out the representation noise as in Equation 4.11. This is likely because in this experiment, the prior assumption – that $\mathcal{D}_1$ and $\mathcal{D}_2$ follow the same distribution – is poor, as discussed in Section 4.2.2.

Classification accuracy results are given in Table 5.2. The classifier trained on CLM datapoints performs at 89.3% accuracy at $\epsilon = 10$, and notably we see performance is well maintained as we increase the privacy guarantee on the training data, performing at 82.8% accuracy at $\epsilon = 1$. This indicates that sufficient utility is retained in our training set for the purposes of solving this downstream task, at every ϵ tested. As before, the CLM significantly outperforms the benchmark mechanisms. The accuracy of the classifier trained on data privatised with PrivUnit (the best performing benchmark) is 20 percentage points lower accuracy than the CLM at $\epsilon = 10$, and only marginally above random accuracy at $\epsilon = 6$, performing at 55.5%.

Table 5.3: Private Accuracy of classifiers trained on ϵ_{train}-LDP (image, label) tuples collected using different LDP mechanisms. ϵ_{test} refers to the LDP guarantee of the images classified at inference time. Error bars represent ±1 standard deviation from the mean over 3 trials.

	Privacy Level	CLM	PrivUnit	Duchi	Laplace	Upper Bound
CIFAR-10	$\epsilon_{\text{TRAIN}} = 10$, $\epsilon_{\text{TEST}} = 7$	**45.0±0.3**	9.6±0.5	10.5±1.2	10.1±0.5	80.7
	$\epsilon_{\text{TRAIN}} = 8$, $\epsilon_{\text{TEST}} = 5.6$	**37.8±0.2**	10.0±0.4	10.0±0.7	10.2±1.3	69.3
	$\epsilon_{\text{TRAIN}} = 6$, $\epsilon_{\text{TEST}} = 4.2$	**27.8±0.5**	9.5±0.6	9.8±0.6	9.7±0.3	53.8
	$\epsilon_{\text{TRAIN}} = 4$, $\epsilon_{\text{TEST}} = 2.8$	**15.9±0.9**	10.1±0.7	9.9±0.7	10.4±0.8	36.0
	$\epsilon_{\text{TRAIN}} = 2$, $\epsilon_{\text{TEST}} = 1.4$	10.3±0.5	9.8±0.9	9.6±0.5	**10.6±0.6**	20.0
	$\epsilon_{\text{TRAIN}} = 1$, $\epsilon_{\text{TEST}} = 0.7$	**10.5±0.5**	10.4±0.8	10.3±0.5	9.8±0.3	14.2
	No Privacy	86.3±0.0	76.1±0.7	76.1±0.7	76.1±0.7	100.0

5.4 Classifying Private Datapoints

Finally, in an experiment that mirrors those conducted in Section 4.4, we test the performance of a classifier that is both trained on LDP datapoints, and aims to classify LDP datapoints at inference time. Acknowledging that a considerable amount of information is lost during privatisation, we compare our results to Equation 4.32, which outlines a proxy to the maximum achievable accuracy of our Laplace mechanism (see Section 4.4.2 for details). This is denoted 'Upper Bound' in the results (Table 5.3). As in Section 4.4.3, ϵ_{train} indicates the privacy guarantee of the (data, label) pairs in the classifier training set, and ϵ_{test} indicates the privacy value of the privatised (unlabelled) test points on which we test our classifier ($\lambda = 70\%$ of ϵ_{train}).

Results show a significant drop in performance when the classifier acts on LDP data. We see that we don't saturate the bound, but again note that the bound assumes we only encode class information into the representation. In addition, the bound assumes we can construct a perfect classifier in the non-private scenario, but our model achieves only 86.3% in practice on CIFAR-10 at $\epsilon = \infty$ (in the VLM this had far less effect since one can achieve very high classification accuracy on MNIST). Despite this, the bound presents a pessimistic result, and we question whether even a mechanism that could saturate the bound would be useful in many practical applications.

5.5 Conclusion

The VLM displayed significant improvements over state-of-the-art LDP mechanisms when applied to data considered high-dimensional in the context of LDP. However, the question still remained as to whether this approach could be scaled up to datasets commonly used in modern machine learning applications. In this chapter, we demonstrated that the fundamental ideas introduced in Chapter 4 can be scaled up to high-dimensional data privatisation under the guarantees of LDP. Furthermore, by introducing simple modifications to arbitrary, existing representation learning algorithms, we have presented a straightforward method for the training of LDP mechanisms that privatise any data modality for which representation learning is possible. This has opened up our approach to real-world applications at scale, with clear implications for the way in which organisations collect sensitive data.

Chapter 6

Conclusion

The first goal of this thesis was to develop a latent variable model that naturally encodes information into the latent variable, irrespective of the choice of generative network. In Chapter 3, we introduced a novel latent variable model to tackle this problem. Rather than maximising a lower bound on the data log likelihood, we instead optimise an objective that encourages both data generation, as in the standard variational autoencoder, and data reconstruction, much like a probabilistic version of the deterministic autoencoder. As with the deterministic autoencoder, learning to reconstruct the input data necessitates that information be encoded into the latent variable. Thus, our model circumvents the phenomenon of 'posterior collapse' that plagues variational autoencoders when a powerful generative model is used. In addition, the objective function of our generative model provides a principled interpretation for the use of ad-hoc annealing factors frequently used in the variational autoencoder objective.

We conducted a series of experiments applied to language modelling, demonstrating the strong performance of our approach with powerful LSTM architectures in the generative model. When applied to such tasks, variational autoencoders typically rely on KL annealing factors in the standard ELBO [Bowman et al., 2016], weakened generative networks [Yang et al., 2017], or aggressive updates to the posterior network [He et al., 2019].

Since the work in Chapter 3 was originally published, large transformer-based models have become the de facto standard for many language modelling tasks [Devlin

et al., 2019, Radford et al., 2019], typically outperforming variational autoencoders on such applications. Despite this, variational autoencoders are still frequently used in applications that necessitate powerful generative distributions, and our approach provides a valuable alternative here.

The second goal of this thesis was to develop machine learning models to act as powerful mechanisms for the privatisation of high-dimensional data, under the strict guarantees of local differential privacy (LDP). While existing work typically utilises fixed LDP mechanisms, we propose using auxiliary data (i.e. data that an organisation has access to either internally or publicly) to *learn* a LDP mechanism. This provides increased control over specific attributes of the mechanism, allowing one to add privacy-inducing noise on a low dimensional manifold, circumventing the so-called 'curse of dimensionality' that plagues existing mechanisms.

As a consequence of this, the dimension of data we are able to privatise, whilst both retaining data utility and maintaining strict privacy guarantees, is orders of magnitude higher than that privatised by existing LDP mechanisms. To our knowledge, Bhowmick et al. [2019] present the only other mechanism designed to privatise data in such high dimensions. However, where they enforce only relatively weak privacy guarantees (testing $\epsilon \geq 50$ experimentally), the research we introduce induces LDP for single-digit ϵ (i.e. $\epsilon \leq 10$). Consequently, our approach provides the best solution yet for organisations looking to collect large scale data without violating data owners' privacy.

We use the collected LDP data as a training set for downstream model training, demonstrating empirically that the utility of our LDP training data surpasses that of training data privatised with state-of-the-art benchmark LDP mechanisms. The training of performant downstream models is a common real-world application for data collectors that was previously thought infeasible when the training data satisfies the guarantees of LDP.

We introduce two different representation learning approaches to learning LDP mechanisms.

In Chapter 4, we develop a novel latent variable model for LDP privatisation. We induce LDP in latent space by constraining the mean of the inference distribution

before adding carefully calibrated noise to the inferred representations. Furthermore, a data reconstruction, obtained by passing the LDP latent representation through the decoder, is equivalent to adding complex non-linear, LDP-inducing noise in the original feature space.

To train the VLM, we optimise a variational lower bound on the log likelihood – the same objective used to train a VAE. In Chapter 3, we discussed how models with powerful generative networks can learn to ignore the latent variable when trained under this objective. In all experiments in Chapter 4, we found the capacity of generative network required to model MNIST and Lending Club was sufficiently small that this was not an issue. However, if we were to privatise data that required a more powerful generative network, such as the language data from Chapter 3, we would advocate using the same constrained, LDP-inducing Laplace inference distribution, but trained with the AutoGen objective (Equation 3.10) rather than the standard lower bound (Equation 4.1). Indeed, if the VLM were to suffer from posterior collapse, our privatised representations would contain little information for downstream learning.

In Chapter 5, we provide a clear and concise framework for the adaptation of a broad array of representation learning models, enabling the privatisation of much higher-dimensional data. To demonstrate this, we develop a mechanism for privatising colour image data by adapting a state-of-the-art contrastive representation learning model. In presenting an approach that adapts existing algorithms, rather than designing LDP mechanisms form the ground up, we are able to capitalise on the myriad advancements over the past few years in the much more mature field of representation learning. All results in Chapter 4 and 5 significantly outperform current state-of-the-art benchmark mechanisms.

We believe such mechanisms could have clear and immediate implications for the practice of data collection. Previously, organisations collecting data for the training of machine learning models attempted to justify privacy violations as a necessary evil, in the absence of practical, high-utility LDP-inducing approaches. However our work demonstrates that it is possible to conduct high-dimensional data collection, obtaining LDP representations suitable for downstream model training,

whilst protecting the privacy of individuals.

The performance of our mechanisms is hinged upon two distinct attributes: the utility of the learnt representation (specifically the representation's effectiveness for downstream learning); and the extent to which inducing LDP noise destroys this utility. Noting this, we highlight several avenues for future research. These include:

- **Choice of LDP mechanism:** The work in Chapters 4 and 5 exclusively uses the local Laplace mechanism to induce LDP. However, extensive work has gone into developing mechanisms that are optimal under certain conditions [Duchi et al., 2018, Bhowmick et al., 2019]. We hypothesise that the use of alternative mechanisms to induce LDP on the low dimensional manifold, or indeed the creation of new mechanisms tailored for privatisation in representation space, may lead to further improvements. For the VLM, we are constrained to mechanisms that induce LDP via reparameterisable noise distributions, but for the CLM we have no such constraints.

- **Encouraging noise robustness:** Though we present an effective approach for encouraging the noise robustness of our learnt, low-dimensional representations, an exploration of alternative techniques to encourage noise-robustness could lead to significant improvements in the retention of data utility after privatisation. For example, there may exist similarity metrics in Equation 5.8 that lead to better noise-robust representations than cosine similarity. Extending this further, one may consider developing a representation learning approach from the ground up designed specifically for LDP privatisation, rather than adapting existing representation learning methods.

- **Privatising other data modalities:** Our mechanisms overcome significant hurdles for high-dimensional data privatisation, and we demonstrated this empirically on tabular and image data. However, there exist a wide range of data modalities for which LDP mechanisms would be valuable. For example, we hypothesise that our mechanism could have particularly strong implications in natural language processing, where large transformer models [Devlin et al., 2019, Yang et al., 2021] have demonstrated not just an ability to learn powerful

representations, but strong performance on transfer learning tasks. A pre-trained LDP mechanism for language could be trained on data scraped from the internet (as in Radford et al. [2019], for example) and released publicly. Such public mechanisms would allow organisations to collect data privately, without requiring access to auxiliary data or the need to train LDP mechanisms.

Appendix A

VLM Experimental Details

A.1 Data Pre-Processing

For every experiment in Chapter 4, we conduct three trials, and calculate the mean and standard deviation of accuracy for each set of trials. The error bars represent one standard deviation above and below the mean.

We use the MNIST and Lending Club datasets. MNIST is a dataset containing 70,000 labelled grayscale images of handwritten digits from 10 classes, corresponding to digits 0-9; the dataset is split into 60,000 training points and 10,000 test points. We convert the images to values between 0 and 1 by dividing each pixel value by 255 and treating them as continuous.

Lending Club is a tabular, financial dataset made up of around 540,000 entries with 23 continuous and categorical features (after pre-processing, before one-hot encoding); the task is binary classification, to determine whether a loan will be repaid. It is split into train, validation and test sets according to the issue date of the loans. The oldest 85% of data forms the training data, with the remaining forming the validation and test data. We perform a number of standard pre-processing steps on this dataset, including:

- Dropping features that contain too many missing values, and those that would not normally be available to a loan company.
- Mean imputation to fill remaining missing values.

- Standard scaling of continuous features. Extreme outliers (those with features more than 10 standard deviations from the mean) are removed here.

- Balancing the target classes by dropping the excess class 0 entries.

- One-hot encoding categorical variables.

A.2 Data Splits

In practice, a data collector would construct a VLM training set from clean auxiliary data that they have access to, either internally or publicly. They would then use this trained VLM to collect further data under LDP guarantees. This collected private data can then be used for downstream tasks. In our experiments, we split the data into two smaller sets to simulate this real-world implementation. The first dataset is used as auxiliary VLM training data $\mathcal{D}_1$; the second dataset simulates data $\mathcal{D}_2$ held by the data owner that is privatised and collected using the VLM, and ultimately used for classifier training. The way in which we split our data differs between experiments in Chapter 4, and we outline each method in the following sections.

Data Collection

Data collection experiments are conducted on both MNIST and Lending Club. For all results in Sections 4.3.1 and 4.4 (except those of the data split ablation in Section 4.3.1.2) we split the data using 75% for VLM training and the remainder for the classifier. The mechanism test set is used for validation since no test set is required here. The classifier training points are split randomly in a 9:1 ratio to form training and validation sets. We report classifier performance on the classifier test set. In Section 4.3.1.2 we conduct an ablation study that explores the extent to which this choice of data split affects classifier performance, and outline data splits in Table 4.3.

Novel-Class Classification

This experiment was conducted on MNIST only. We use a similar approach to the above, but split the data between the mechanism and the classifier such that the mechanism train/validation sets contain $\frac{8}{9}$ths of (unlabelled) training images from classes 0 to 8. The remaining $\frac{1}{9}$th of 0 to 8 images, and all 9s, are used for classifier train, test and validation sets. Our mechanism datasets then contain equal class balance for the classes 0 to 8, and the classifier datasets contain equal class balance for 9s and 'not 9s'.

Data Join

This experiment was conducted on Lending Club only. The dataset is split column-wise between the 23 features, such that 8 features remain non-privatised (month of earliest credit line, number of open credit lines, initial listing status of loan, application type, address (US state), home ownership status, employment length, public record bankruptcies) and the remaining 15 features are privatised. This feature split was chosen such that the 8 non-private features contain some information to solve the classification task, while the remaining 15 features contain information which, at least before privatisation, further improves classifier performance.

A.3 Benchmarks

We study the local Laplace mechanism, the mechanism introduced in Duchi et al. [2018] (which we refer to as "Duchi's mechanism"), the Hybrid mechanism [Wang et al., 2019], and the PrivUnit$_2$ mechanism [Bhowmick et al., 2019].

For the Laplace mechanism, we privatise each feature independently, choosing the noise level for each of the d features such that $\epsilon_i = \epsilon/d$. We do this since we have no prior knowledge about which features are most important. We then assume Δf_i is equal to the difference between the maximum and minimum value of the feature i within the training and validation sets used to train the mechanism in the main experiments, after pre-processing. We then have to clip any values that lie outside

Table A.1: VLM hyperparameters used for data join experiments on the Lending Club dataset.

ϵ	d	l	$\epsilon_{\text{PRE-TRAINING}}$
∞	8	5	20
10	8	5	10
8	5	5	15
6	5	5	15
4	5	5	10
2	5	5	15
1	5	5	10

this interval in the collected dataset at privatisation time. For tabular experiments, we privatise categorical features using a flip mechanism.

We omit the hybrid mechanism (Algorithm 4 of Wang et al. [2019]) since it entails collecting only $k \leq 4$ of the d features for our experiments, but implement Duchi's mechanism as in Algorithm 3 of Wang et al. [2019]. Again, we privatise categorical features using a flip mechanism.

For PrivUnit$_2$, we treat each image as a d-dimensional vector and privatise both its direction and magnitude, as outlined in the PrivUnit$_2$ and ScalarDP algorithms in Bhowmick et al. [2019]. We omit Lending Club experiments, since PrivUnit$_2$ is not designed for mixed-type data.

For all tasks, we used feedforward architectures for our benchmark classifiers. More powerful convolutional architectures were not found to improve performance.

A.4 Hyperparameter Choices

In order to find the optimal experimental setup, we conduct a private grid search over a number of the hyperparameters in our model, as outlined in Section 4.2.4. We searched over the following model hyperparameters:

- The proportion λ of our privacy budget assigned to the representation vs. the label i.e. $\lambda = \epsilon_x/(\epsilon_x + \epsilon_y)$.

- The ℓ_1 clipping distance l of our inference network mean i.e. $l = \Delta f_\phi/2$.

Table A.2: DP-Adam hyperparameters used for the VLM data collection experiments under CDP.

Task	ϵ_{CDP}	Learning Rate	Batch Size	Noise Multiplier
MNIST	5	5e-4	64	0.7
	1	5e-4	64	1.1
Lending Club	5	1e-4	128	0.56
	1	1e-4	128	1.1

- The Laplace distribution scale b of our approximate posterior distribution during VLM training. We report this in terms of the $\epsilon_{\text{pre-training}}$-LDP value induced by this Laplace noise i.e. $\epsilon_{\text{pre-training}} = 2l/b$. This is fixed throughout training, unless 'learnt' is specified in Table A.3, which indicates the parameter b is a learnt scalar in the VLM.

- The representation dimension d. We fixed $d = 8$ for MNIST. For Lending Club, we fixed $d = 8$ for the data collection experiments but searched over $d \in \{5, 8\}$ for the data join experiments due to the smaller number of features.

For VLM training, we use a learning rate of 10^{-4} and batch size of 128 for Lending Club experiments, and we use a learning rate of 5×10^{-4} and batch size of 64 for MNIST. For CDP training experiments in Section 4.3.1, we also did a private grid search over values for the noise multiplier, batch size, and DP learning rate for central $\epsilon \in \{1, 5\}$. The DP-Adam [Gylberth et al., 2017] hyperparameter max gradient norm was fixed to 1 throughout. The number of training epochs needed to reach the target central ϵ value follows from the choice of hyperparameters, combined with the VLM training set size (45,000 for MNIST, and 341,000 for Lending Club). Note that we fixed $\delta = 10^{-5}$ for the CDP guarantee in all experiments.

The results from these grid searches are given in Tables A.1, A.2, and A.3.

Table A.3: VLM hyperparameters used for the data collection and novel-class classification experiments.

Experiment	Task	ϵ	λ	d	l	$\epsilon_{\text{pre-training}}$
Classifying clean datapoints (Section 4.3 experiments)	MNIST	∞	N/A	8	10	Learnt
		10	0.7	8	10	33
		8	0.7	8	5	32
		6	0.7	8	5	19
		4	0.7	8	7.5	13
		2	0.7	8	7.5	7
		1	0.7	8	5	7
	Lending Club	∞	N/A	8	10	Learnt
		10	0.7	8	5	15
		8	0.7	8	5	29
		6	0.7	8	5	29
		4	0.7	8	5	15
		2	0.95	8	5	15
		1	0.95	8	10	21
Classifying LDP datapoints (Section 4.4 experiments)	MNIST	∞	N/A	8	10	Learnt
		10	0.7	8	10	5
		8	0.7	8	7.5	5
		6	0.7	8	7.5	5
		4	0.7	8	5	5
		2	0.7	8	5	15
		1	0.7	8	7.5	15

A.5 Mechanism Architectures and Transformations

For MNIST, we use a VLM encoder network with 3 hidden layers of size {400, 150, 50}, and a decoder network with 3 hidden layers of size {50, 150, 400}. For Lending Club, we use a VLM encoder and decoder network with 2 hidden layers of size {500, 500}. For the classifier, we used a a network with 1 hidden layer of size 50.

Appendix B

CLM Experimental Details

B.1 Data Pre-Processing

As in Chapter 4, we conduct three trials for each experiment and calculate the mean and standard deviation of accuracy for each set of trials. The error bars represent one standard deviation above and below the mean.

We use the CIFAR-10 dataset: a colour image dataset containing 60,000 images from 10 classes. We convert the images to values between 0 and 1 by dividing each pixel value by 255 and treating them as continuous.

As before, for data collection experiments, we split both sets using 75% for mechanism training and the remainder for classifier training. The mechanism test set is used for validation since no test set is required here. The classifier training points are split randomly in a 9:1 ratio to form training and validation sets. We report classifier performance on the classifier test set.

For novel class classification we split the data between the mechanism and the classifier such that the mechanism train/validation sets contain (unlabelled) training images from classes 0 to 8, and the classifier datasets contain equal class balance for 9s and 'not 9s', and described in Section A.2.

We emphasise that these data splits are designed to simulate real world scenarios. In reality, the data split would be pre-determined, with the mechanism training set comprised of clean data the data collector has access to, either internally or publicly. The data collector would use this trained mechanism to collect data under LDP.

Table B.1: CLM hyperparameters used for the data collection and novel-class classification experiments.

Experiment	Task	ϵ	λ	l	$\epsilon_{\text{pre-training}}$
Classifying clean datapoints (Section 5.3 experiments)	CIFAR-10	∞	N/A	5	No noise
		10	0.7	5	No noise
		8	0.7	5	No noise
		6	0.7	5	70
		4	0.7	5	30
		2	0.7	5	20
		1	0.7	5	20
Classifying LDP datapoints (Section 5.4 experiments)	CIFAR-10	∞	N/A	5	No noise
		10	0.7	5	27
		8	0.7	5	24
		6	0.7	5	20
		4	0.7	5	15
		2	0.7	5	8
		1	0.7	5	8

B.2 Benchmarks

We study the same set of benchmarks as in Chapter 4: the local Laplace mechanism, Duchi's mechanism [Duchi et al., 2018], the Hybrid mechanism [Wang et al., 2019], and the PrivUnit$_2$/ ScalarDP mechanism [Bhowmick et al., 2019]. The implementation of these benchmarks is the same as that discussed in Section A.3, with the exception that we experiment with using both a pre-trained and a randomly initialised ResNet-18 for classification, as is used in the CLM encoder. Ultimately however, we found a feedforward architecture with hidden layers of dimension {400, 150, 50} achieved better accuracy than either ResNet-18 model at all local-ϵ values (except local-$\epsilon = \infty$, where instead a pre-trained ResNet-18 architecture was used).

B.3 Hyperparameter Choices

For the CLM training we use a learning rate of 3×10^{-4} and a batch size of 128 (the largest our GPU would allow). We fixed the representation dimension d to 32. As in Chapter 4, we searched over the following model hyperparameters:

- The division λ of our privacy budget between the representation and the label

i.e. $\lambda = \epsilon_x/(\epsilon_x + \epsilon_y)$.

- The ℓ_1 clipping distance l the encoder $f_\phi(\cdot)$ i.e. $l = \Delta f_\phi/2$.
- The scale b of the zero mean Laplace noise added during pre-training of the CLM. This is reported in terms of the $\epsilon_{\text{pre-training}}$-LDP value induced by this Laplace noise i.e. $\epsilon_{\text{pre-training}} = 2l/b$. 'No privacy' in Table B.1 indicates we set $b = 0$.

The results from this grid search are given in Table B.1.

Bibliography

M. Abadi, A. Chu, I. Goodfellow, H. B. McMahan, I. Mironov, K. Talwar, and L. Zhang. Deep learning with differential privacy. In *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*, 2016.

G. Acs, L. Melis, C. Castelluccia, and E. De Cristofaro. Differentially private mixture of generative neural networks. *IEEE Transactions on Knowledge and Data Engineering*, 2019.

Amnesty International. Surveillance giants: How the business model of Google and Facebook threatens human rights, 2019.

J. Ballé, D. Minnen, S. Singh, S. J. Hwang, and N. Johnston. Variational image compression with a scale hyperprior. In *International Conference on Learning Representations*, 2018.

B. B. Bhattacharya. Maximizing Voronoi regions of a set of points enclosed in a circle with applications to facility location. *Journal of Mathematical Modelling and Algorithms*, 2010.

A. Bhowmick, J. Duchi, J. Freudiger, G. Kapoor, and R. Rogers. Protection against reconstruction and its applications in private federated learning. *CoRR*, abs/1812.00984, 2019.

S. R. Bowman, L. Vilnis, O. Vinyals, A. Dai, R. Jozefowicz, and S. Bengio. Generating sentences from a continuous space. In *Conference on Computational Natural Language Learning*, 2016.

K. Chatzikokolakis, M. E. Andrés, N. E. Bordenabe, and C. Palamidessi. Broadening the scope of differential privacy using metrics. In *Privacy Enhancing Technologies Symposium*, 2013.

Q. Chen, C. Xiang, M. Xue, B. Li, N. Borisov, D. Kaafar, and H. Zhu. Differentially private data generative models. *CoRR*, abs/1812.02274, 2018.

T. Chen, S. Kornblith, M. Norouzi, and G. Hinton. A simple framework for contrastive learning of visual representations. In *International Conference on Machine Learning*, 2020.

R. Child. Very deep VAEs generalize autoregressive models and can outperform them on images. In *International Conference on Learning Representations*, 2021.

Competition and Markets Authority. Online platforms and digital advertising market study, Appendix F: the role of data in digital advertising, 2020.

A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society: Series B (Methodological)*, 1977.

J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. ImageNet: A large-scale hierarchical image database. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2009.

E. Denton and R. Fergus. Stochastic video generation with a learned prior. In *International Conference on Machine Learning*, 2018.

J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics*, 2019.

A. B. Dieng, C. Wang, J. Gao, and J. Paisley. TopicRNN: A recurrent neural network with long-range semantic dependency. In *International Conference on Learning Representations*, 2017.

B. Ding, J. Kulkarni, and S. Yekhanin. Collecting telemetry data privately. In *Advances in Neural Information Processing Systems*, 2017.

J. Donahue, L. A. Hendricks, S. Guadarrama, M. Rohrbach, S. Venugopalan, K. Saenko, and T. Darrell. Long-term recurrent convolutional networks for visual recognition and description. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2015.

J. Duchi, E. Hazan, and Y. Singer. Adaptive subgradient methods for online learning and stochastic optimization. *Journal of Machine Learning Research*, 2011.

J. C. Duchi, M. I. Jordan, and M. J. Wainwright. Local privacy and statistical minimax rates. In *IEEE Annual Symposium on Foundations of Computer Science*, 2013.

J. C. Duchi, M. I. Jordan, and M. J. Wainwright. Minimax optimal procedures for locally private estimation. In *Journal of the American Statistical Association*, 2018.

C. Dwork and A. Roth. The algorithmic foundations of differential privacy. In *Foundations and Trends in Theoretical Computer Science*, 2014.

C. Dwork, F. McSherry, K. Nissim, and A. Smith. Calibrating noise to sensitivity in private data analysis. In *Theory of Cryptography*, 2006.

Ú. Erlingsson, V. Pihur, and A. Korolova. RAPPOR: Randomized aggregatable privacy-preserving ordinal response. In *Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security*, 2014.

G. Fanti, V. Pihur, and Ú. Erlingsson. Building a RAPPOR with the unknown: Privacy-preserving learning of associations and data dictionaries. *Proceedings on Privacy Enhancing Technologies*, 2016.

M. Fredrikson, S. Jha, and T. Ristenpart. Model inversion attacks that exploit confidence information and basic countermeasures. In *ACM SIGSAC Conference on Computer and Communications Security*, 2015.

C. Frye, D. de Mijolla, T. Begley, L. Cowton, M. Stanley, and I. Feige. Shapley explainability on the data manifold. In *International Conference on Learning Representations*, 2021.

X. Glorot, A. Bordes, and Y. Bengio. Deep sparse rectifier neural networks. In *International Conference on Artificial Intelligence and Statistics*, 2011.

A. Goyal, A. Lamb, Y. Zhang, S. Zhang, A. Courville, and Y. Bengio. Professor forcing: A new algorithm for training recurrent networks. In *Advances in Neural Information Processing Systems*, 2016.

A. Graves. Generating sequences with recurrent neural networks. *CoRR*, abs/1308.0850, 2013.

A. Graves and N. Jaitly. Towards end-to-end speech recognition with recurrent neural networks. In *International Conference on Machine Learning*, 2014.

I. Gulrajani, K. Kumar, F. Ahmed, A. A. Taiga, F. Visin, D. Vazquez, and A. Courville. PixelVAE: A latent variable model for natural images. In *International Conference on Learning Representations*, 2017.

R. Gylberth, R. Adnan, S. Yazid, and T. Basaruddin. Differentially private optimization algorithms for deep neural networks. In *International Conference on Advanced Computer Science and Information Systems*, 2017.

R. Habib, S. Mariooryad, M. Shannon, E. Battenberg, R. Skerry-Ryan, D. Stanton, D. Kao, and T. Bagby. Semi-supervised generative modeling for controllable speech synthesis. In *International Conference on Learning Representations*, 2020.

J. He, D. Spokoyny, G. Neubig, and T. Berg-Kirkpatrick. Lagging inference networks and posterior collapse in variational autoencoders. *International Conference on Learning Representations*, 2019.

K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2016.

K. He, H. Fan, Y. Wu, S. Xie, and R. Girshick. Momentum contrast for unsupervised visual representation learning. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2020.

I. Higgins, L. Matthey, A. Pal, C. Burgess, X. Glorot, M. Botvinick, S. Mohamed, and A. Lerchner. beta-VAE: Learning Basic Visual Concepts with a Constrained Variational Framework. In *International Conference on Learning Representations*, 2017.

G. E. Hinton and R. R. Salakhutdinov. Reducing the Dimensionality of Data with Neural Networks. *Science*, 313(5786):504–507, 2006.

G. E. Hinton and R. S. Zemel. Autoencoders, Minimum Description Length and Helmholtz Free Energy. In *Advances in Neural Information Processing Systems*, 1994.

S. Hochreiter and J. Schmidhuber. Long short-term memory. *Neural Computation*, 9 (8):1735–1780, 1997.

K. Hornik. Approximation capabilities of multilayer feedforward networks. *Neural Networks*, 1991.

J. Hsu, M. Gaboardi, A. Haeberlen, S. Khanna, A. Narayan, B. Pierce, and A. Roth. Differential privacy: an economic method for choosing epsilon. In *IEEE Computer Security Foundations Symposium*, 2014.

Institute for Critical Infrastructure Technology. Hacking healthcare in 2016: Lessons the healthcare industry can learn from the OPM breach, 2016.

J. A. Irvin, P. Rajpurkar, M. Ko, Y. Yu, S. Ciurea-Ilcus, C. Chute, H. Marklund, B. Haghgoo, R. L. Ball, K. Shpanskaya, J. Seekins, D. Mong, S. Halabi, J. Sandberg, R. Jones, D. Larson, C. Langlotz, B. Patel, M. Lungren, and A. Ng. CheXpert: A large chest radiograph dataset with uncertainty labels and expert comparison. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2019.

D. Jamieson and M. Loverde. Position-dependent Voronoi probability distribution functions for matter and halos. *Physics Review D*, 2021.

K. Jarrett, K. Kavukcuoglu, M. Ranzato, and Y. LeCun. What is the best multi-stage architecture for object recognition? In *IEEE International Conference on Computer Vision*, 2009.

J. Jolly. FCA admits revealing confidential details of 1,600 consumers. In *The Guardian*, 2020.

S. P. Kasiviswanathan, H. K. Lee, K. Nissim, S. Raskhodnikova, and A. Smith. What can we learn privately? In *IEEE Symposium on Foundations of Computer Science*, 2008.

D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. In *International Conference on Learning Representations*, 2015.

D. P. Kingma and M. Welling. Auto-encoding variational Bayes. In *International Conference on Learning Representations*, 2014.

D. P. Kingma, S. Mohamed, D. J. Rezende, and M. Welling. Semi-supervised learning with deep generative models. In *Advances in Neural Information Processing Systems*, 2014.

F. H. Kingma, P. Abbeel, and J. Ho. Bit-swap: Recursive bits-back coding for lossless compression with hierarchical latent variables. In *International Conference on Machine Learning*, 2019.

S. Krehbiel. Choosing epsilon for privacy as a service. *Proceedings on Privacy Enhancing Technologies*, 2019.

A. Krizhevsky. Learning multiple layers of features from tiny images, 2009.

Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 1998.

J. Lee and C. Clifton. How much is enough? choosing ϵ for differential privacy. In *Information Security*, 2011.

J. Lee, W. Yoon, S. Kim, D. Kim, S. Kim, C. H. So, and J. Kang. BioBERT: a pre-trained biomedical language representation model for biomedical text mining. *Bioinformatics*, 2019.

Y. Lee, J. Shin, and K. Jung. Bidirectional variational inference for non-autoregressive text-to-speech. In *International Conference on Learning Representations*, 2021.

F. Locatello, S. Bauer, M. Lucic, G. Raetsch, S. Gelly, B. Scholkopf, and O. Bachem. Challenging common assumptions in the unsupervised learning of disentangled representations. In *International Conference on Machine Learning*, 2019.

L. Maaløe, M. Fraccaro, V. Liévin, and O. Winther. BIVA: A very deep hierarchy of latent variables for generative modeling. In *Advances in Neural Information Processing Systems*, 2019.

A. Machanavajjhala, J. Gehrke, D. Kifer, and M. Venkitasubramaniam. L-diversity: privacy beyond k-anonymity. In *International Conference on Data Engineering*, 2006.

D. Mahajan, R. Girshick, V. Ramanathan, K. He, M. Paluri, Y. Li, A. Bharambe, and L. van der Maaten. Exploring the limits of weakly supervised pretraining. In *Proceedings of the European Conference on Computer Vision*, 2018.

A. Mansbridge, R. Fierimonte, I. Feige, and D. Barber. Improving latent variable descriptiveness by modelling rather than ad-hoc factors. *Machine Learning (ECML PKDD Journal Track)*, 2019.

A. Mansbridge, G. Barbour, D. Piras, M. Murray, C. Frye, I. Feige, and D. Barber. Representation learning for high-dimensional data collection under local differential privacy. *CoRR*, abs/2010.12464, 2022.

G. Martin, P. Martin, C. Hankin, A. Darzi, and J. Kinross. Cybersecurity and healthcare: how safe are we? *British Medical Journal*, 2017.

L. Mehner, S. Nuñez von Voigt, and F. Tschorsch. Towards explaining epsilon: A worst-case study of differential privacy risks. In *IEEE European Symposium on Security and Privacy Workshops*, 2021.

A. Narayanan and V. Shmatikov. How to break anonymity of the netflix prize dataset. *CoRR*, abs/cs/0610105, 2007.

Y. Nesterov. A method for solving the convex programming problem with convergence rate O($1/k^2$). *Dokl. akad. nauk Sssr 269*, 269:543–547, 1983.

J. Paisley, D. Blei, and M. Jordan. Variational bayesian inference with stochastic search. In *International Conference on Machine Learning*, 2012.

N. Papernot, N. Abadi, U. Erlingsson, I. Goodfellow, and K. Talwar. Semi-supervised knowledge transfer for deep learning from private training data. In *International Conference on Learning Representations*, 2017.

A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, and I. Sutskever. Language models are unsupervised multitask learners, 2019.

A. Ramesh, M. Pavlov, G. Goh, S. Gray, C. Voss, A. Radford, M. Chen, and I. Sutskever. Zero-shot text-to-image generation. *CoRR*, abs/2102.12092, 2021.

R. Reddy. Speech understanding systems: summary of results of the five-year research effort at Carnegie-Mellon University. In *Technical Report, Carnegie-Mellon University*, 1977.

X. Ren, C. Yu, W. Yu, S. Yang, X. Yang, J. A. McCann, and P. S. Yu. LoPub: High-dimensional crowdsourced data publication with local differential privacy. In *IEEE Transactions on Information Forensics and Security*, 2018.

D. J. Rezende, S. Mohamed, and D. Wierstra. Stochastic backpropagation and approximate inference in deep generative models. In *International Conference on Machine Learning*, 2014.

E. Riol, J. C. Puche, F. J. Delgado, J. Finat, and R. Martinez. Weighted Voronoi diagrams for optimal location of goods and services in planar maps. In *International Symposium on Voronoi Diagrams in Science and Engineering*, 2011.

D. E. Rumelhart, G. E. Hinton, and R. J. Williams. Learning representations by back-propagating errors. *Nature*, 323:533–536, 1986.

T. Salimans, A. Karpathy, X. Chen, and D. P. Kingma. PixelCNN++: A PixelCNN implementation with discretized logistic mixture likelihood and other modifications. In *International Conference on Learning Representations*, 2017.

D. Schmidt. Google data collection, 2018.

S. Semeniuta, A. Severyn, and E. Barth. A Hybrid Convolutional Variational Autoencoder for Text Generation. In *Conference on Empirical Methods in Natural Language Processing*, 2017.

H. Shah, B. Zheng, and D. Barber. Generating sentences using a dynamic canvas. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2018.

C. K. Sønderby, T. Raiko, L. Maaløe, S. K. Sønderby, and O. Winther. Ladder variational autoencoders. In *Advances in Neural Information Processing Systems*, 2016.

I. Sutskever, O. Vinyals, and Q. V. Le. Sequence to sequence learning with neural networks. In *Advances in Neural Information Processing Systems*, 2014.

L. Sweeney. K-anonymity: A model for protecting privacy. *International Journal on Uncertainty, Fuzziness and Knowledge-based Systems*, 2002.

C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna. Rethinking the inception architecture for computer vision. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2016.

S. Takagi, T. Takahashi, Y. Cao, and M. Yoshikawa. P3GM: Private high-dimensional data release via privacy preserving phased generative model. In *IEEE International Conference on Data Engineering*, 2021.

B. Thomee, D. A. Shamma, G. Friedland, B. Elizalde, K. Ni, D. Poland, D. Borth, and L.-J. Li. YFCC100M: The new data in multimedia research. In *Association for Computing Machinery*, 2016.

J. Townsend, T. Bird, and D. Barber. Practical Lossless Compression with Latent Variables using Bits Back Coding. In *International Conference on Learning Representations*, 2019.

A. Triastcyn and B. Faltings. Generating artificial data for private deep learning. In *Proceedings of the PAL: Privacy-Enhancing Artificial Intelligence and Language Technologies, AAAI Spring Symposium Series*, 2019.

A. van den Oord, O. Vinyals, and K. Kavukcuoglu. Neural discrete representation learning. In *Advances in Neural Information Processing Systems*, 2017.

A. van den Oord, Y. Li, and O. Vinyals. Representation learning with contrastive predictive coding. *CoRR*, abs/1807.03748, 2018.

R. Vedantam, I. Fischer, J. Huang, and K. Murphy. Generative Models of Visually Grounded Imagination. In *International Conference on Learning Representations*, 2018.

N. Wang, X. Xiao, Y. Yang, J. Zhao, S. C. Hui, H. Shin, J. Shin, and G. Yu. Collecting and analyzing multidimensional data with local differential privacy. In *IEEE International Conference on Data Engineering*, 2019.

T. Wang, J. Zhao, Z. Hu, X. Yang, X. Ren, and K.-Y. Lam. Local differential privacy for data collection and analysis. *Neurocomputing*, 2021.

Y. Wang, X. Wu, and D. Hu. Using randomized response for differential privacy preserving data collection. In *Technical Report, DPL-2014-003, University of Arkansas*, 2014.

S. L. Warner. Randomized response: A survey technique for eliminating evasive answer bias. *Journal of the American Statistical Association*, 1965.

R. J. Williams and D. Zipser. A Learning Algorithm for Continually Running Fully Recurrent Neural Networks. *Neural Computation*, 1989.

M. Wu and N. Goodman. Multimodal Generative Models for Scalable Weakly-Supervised Learning. *CoRR*, abs/1802.05335, 2018.

L. Xie, K. Lin, S. Wang, F. Wang, and J. Zhou. Differentially private generative adversarial network. *CoRR*, abs/1802.06739, 2018.

Z. Yang, Z. Hu, R. Salakhutdinov, and T. Berg-Kirkpatrick. Improved Variational Autoencoders for Text Modeling using Dilated Convolutions. In *International Conference on Machine Learning*, 2017.

Z. Yang, Y. Yang, D. Cer, J. Law, and E. Darve. Universal sentence representation learning with conditional masked language model. In *Conference on Empirical Methods in Natural Language Processing*, 2021.

F. Yu, S. E. Fienberg, A. B. Slavkovic, and C. Uhler. Scalable privacy-preserving data sharing methodology for genome-wide association studies. *Journal of Biomedical Informatics*, 2014.

J. Zhang, G. Cormode, C. M. Procopiuc, D. Srivastava, and X. Xiao. PrivBayes: Private data release via Bayesian networks. *ACM Transactions on Database Systems*, 2017.

L. Zhu, Z. Liu, and S. Han. Deep leakage from gradients. In *Advances in Neural Information Processing Systems*, 2019.

Y. Zhu, R. Kiros, R. Zemel, R. Salakhutdinov, R. Urtasun, A. Torralba, and S. Fidler. Aligning Books and Movies: Towards Story-like Visual Explanations by Watching Movies and Reading Books. In *International Conference on Computer Vision*, 2015.