We also present the robustness of our two proposed approaches with universities, for example, data-driven disease diagnosis studies [19, 30]. The universities may only possess a small amount of medical data to the public for use, but wants to share the data for training an effective machine learning model. Under this scenario, instead of publishing the medical data directly, the hospital could locally use the medical data to train a representation learning model and then publish it. Any university interested in researching disease diagnosis independently can use the representation learning model to convert their small amounts of medical data into a better representation, boosting learning efficiency. Another motivating example is two companies that want to collaborate on a data intelligence task. A data-rich company $A$ may wish to aid a company $B$ in developing a model that helps maximize revenue, but is unwilling or legally unable to share its data with $B$ directly due to its sensitive nature. Again, the company $A$ can train a representation learning model on its large dataset and share it with the company $B$.

The second approach, termed DP-VaeGM, is to share data, where we use the shared data for training a generative model which learns the distribution of the data, and then the generative model is used to generate a new dataset for usage and the new dataset can be shared further. More specifically, we choose to use the variational autoencoder (VAE) [21] as the generative model. Similar to DP-AuGM, we train the VAE by adding differential privacy noise to the gradient. We evaluate the performance of our two proposed approaches across various differential privacy budgets.

In this paper, we study two different approaches to enable data sharing for learning tasks while preserving data privacy. The first approach is to share representation learning models with multiple parties, for which we choose to use a differentially private autoencoder-based generative model (DP-AuGM). The second approach is to share generated data with multiple parties through generative models, for which we choose to use a differentially private variational autoencoder-based generative model (DP-VaeGM). To achieve differential privacy, we train both models by adding differential privacy noise to the gradient. We evaluate the performance of our two proposed approaches across various differential privacy budgets. We also present the robustness of our two proposed approaches against model inversion attacks [15], membership inference attacks [32], and generative adversarial network (GAN) based attacks against collaborative deep learning [20] only in the extended version of this paper available at https://arxiv.org/pdf/1812.02274.pdf.

1 INTRODUCTION

In this paper, we study two different approaches to enable data sharing for learning tasks while preserving data privacy. We aim to protect the data privacy against the state-of-the-art attacks, namely model inversion attacks [15], membership inference attacks [32], and generative adversarial network (GAN) based attacks against collaborative deep learning [20].

The first approach, termed DP-AuGM, is to share models, where we encode the information of the data into a machine learning model for learning data representations and then share the machine learning model instead. This approach is motivated by representation learning [6], which generally aims to use machine learning models to learn a good representation of the data. Then, the models are used to convert data from its raw format into a better representation, thus helping boost the learning efficiency. An example of this representation learning is word2vec [26]. Analogous to natural language processing, in our paper, we choose to use autoencoders [34] for our representation learning model, as these models are commonly used for extracting key features of data. In order to prevent the attackers from inferring sensitive information from the representation learning model, we add differential privacy noise to the training of the representation learning model [4].

As motivation, consider a hospital not allowed to release its medical data to the public for use, but wants to share the data with universities, for example, data-driven disease diagnosis studies [19, 30]. The universities may only possess a small amount of data, such as public medical datasets [1, 3] which are not adequate for training an effective machine learning model. Under this scenario, instead of publishing the medical data directly, the hospital could locally use the medical data to train a representation learning model and then publish it. Any university interested in researching disease diagnosis independently can use the representation learning model to convert their small amounts of medical data into a better representation, boosting learning efficiency. Another motivating example is two companies that want to collaborate on a data intelligence task. A data-rich company $A$ may wish to aid a company $B$ in developing a model that helps maximize revenue, but is unwilling or legally unable to share its data with $B$ directly due to its sensitive nature. Again, the company $A$ can train a representation learning model on its large dataset and share it with the company $B$.

Related work. Most privacy-preserving or secure data-sharing systems use cryptographic or statistical techniques to enable sensitive data protection and sharing [9, 16, 17]. These systems are generally designed as either centralized (e.g., CryptDB [28] and Mona [24]) or decentralized [27]. Unlike previously proposed techniques, the proposed approaches achieve the following three goals: protect the privacy of training data; enable users to locally customize the privacy preference by configuring the generative models; retain the high utility for generated data. The proposed approaches achieve these goals at a lower computational cost than the aforementioned systems.

Differentially Private Data Sharing: Sharing Models versus Sharing Data

Qingrong Chen¹, Chong Xiang², Minhui Xue³, Bo Li¹, Nikita Borisov¹, Dali Kaafar⁴, Haojin Zhu²
¹University of Illinois Urbana-Champaign, USA
²Shanghai Jiao Tong University, China
³The University of Adelaide, Australia
⁴Macquarie University and CSIRO-Data61, Australia

ABSTRACT

In this paper, we study two different approaches to enable data sharing for learning tasks while preserving data privacy. The first approach is to share representation learning models with multiple parties, for which we choose to use a differentially private autoencoder-based generative model (DP-AuGM). The second approach is to share generated data with multiple parties through generative models, for which we choose to use a differentially private variational autoencoder-based generative model (DP-VaeGM). To achieve differential privacy, we train both models by adding differential privacy noise to the gradient. We evaluate the performance of our two proposed approaches across various differential privacy budgets.
differentially private paradigms [5, 12, 14, 31] and cryptographic
techniques such as homomorphic encryption [17].

2 BACKGROUND

2.1 Differential Privacy

Definition 1 ((ε, δ)-Differential Privacy [13]). A randomized
algorithm \( \mathcal{A} : \mathcal{D} \rightarrow \mathcal{R} \) with domain \( \mathcal{D} \) and range \( \mathcal{R} \), is (ε, δ)-
differentially private if for any two adjacent training datasets \( d, d' \subseteq \mathcal{D} \), which differ by at most one training point, and any subset of
outputs \( S \subseteq \mathcal{R} \), it satisfies that:

\[
\Pr[\mathcal{A}(d) \in S] \leq e^\varepsilon \Pr[\mathcal{A}(d') \in S] + \delta.
\]

The parameter \( \varepsilon \) is often called a privacy budget: a smaller budget
yields stronger privacy guarantees. The second parameter \( \delta \) is a failure rate that is not tolerated by the privacy bound defined by \( \varepsilon \).

2.2 Representation Learning

Representation learning aims to automatically extract the key fea-
tures from the input data and a good representation of the data
usually leads to the success of further classification tasks [6].

Autoencoder. An autoencoder is a widely used unsupervised learn-
ing model for representation learning in many scenarios, such as
goal is to learn a representation of data, typically for the purpose
of dimensionality reduction [18, 33, 34]. It simultaneously trains
an encoder, which transforms a high-dimensional data point to a
low-dimensional representation, and a decoder, which reconstructs
a high-dimensional data point from the representation, while try-
ing to minimize the \( l_2 \) norm distance \( \mathcal{L}_2 \) between the original and
reconstructed data. Through this process, the autoencoder is able to
discard irrelevant features and enhance the performance of machine
learning models when facing high-dimensional input data.

2.3 Variational Autoencoder

Resembling the autoencoder, a variational autoencoder (VAE) also
comprises two parts: the encoder and the decoder [21, 29] with a
latent variable \( z \) sampled from a prior distribution \( p(z) = \text{noise} \).
Different from the autoencoder of which the encoder only tries to
reduce the data into lower dimensions, the encoder inside VAE tries
to encode the input data into a Gaussian probability density do-
main [21]. Mathematically, the encoder approximates \( q(z|x) \), which is
also a neural network (encoder), with input \( z \) conditioned on the
data \( x \). Then, a representation of the data will be sampled based on
the output from the encoder. Finally, the decoder tries to recon-
struct a data point based on sampled noise, which approximates the
posterior \( p(x|z) \). The two neural networks, encoder and decoder,
are trained to maximize a lower bound of the log-likelihood of the
data logarithm of the encoder:

\[
\mathbb{E}_{q(z|x)}[\log p(x|z)] - \text{KL}(q(z|x)||p(z)),
\]

where KL is the Kullback-Leibler divergence [10].

Sampling from the VAE is achieved by sampling from the (typ-
ically Gaussian) prior \( p(z) \) and passing the samples through the
decoder network.

3 DIFFERENTIALLY PRIVATE DATA

SHARING

3.1 Data Sharing through Sharing Models (The

Case of DP-AuGM)

We propose the first approach that shares data through sharing of
the representation learning model, autoencoder, to protect privacy
of the shared data while retaining high utility for machine learning
usage (see overview in Figure 1).

For DP-AuGM, we first train an autoencoder with the shared
data using a differentially private training algorithm. We then pub-
lish the encoder and drop the decoder. Next, users feed their raw
data into this encoder to obtain better data representations which
help boost their learning efficiency. Later, these data with new rep-
resentations could be used to train the targeted learning systems in
the future with privacy guarantees for the shared data. We adopt the
deep learning with differential privacy (DP-DL) algorithm [4]
to train the representation learning model autoencoder. DP-DL [4]
achieves differential privacy by injecting random noise in a stochas-
tic gradient descent (SGD) algorithm. At each step of SGD, DP-DL
computes the gradient for a random subset of training points, fol-
lowed by clipping, averaging out each gradient, and adding noise
in order to protect privacy. The algorithm of DP-AuGM is outlined
in Algorithm 1.

DP Analysis for DP-AuGM. In this paper, we adopt the training
algorithm by Abadi et al. [4] to achieve differential privacy. Based
on the moments accountant technique applied in [4], we obtain that
the training algorithm is \( (O(qe\sqrt{T}), \delta) \)-differentially private, where
\( T \) is the number of training steps, \( q \) is the sampling probability, and
(\( \varepsilon, \delta \)) denotes the privacy budget [4]. In addition, we will prove
that any machine learning model which is trained on the data fed
into DP-AuGM, is also differentially private w.r.t. the shared data
and shares the same privacy bound. This also shows the benefit of
sharing a representation learning model: we only need to train one
representation learning model and all the machine learning models
trained over the data from the representation learning model are
differentially private w.r.t. the shared data.

3.2 Data Sharing through Sharing Generated

Data (The Case of DP-VaeGM)

We propose the second approach that shares data via building a
generative model and sharing a new dataset from the generative
model (see overview in Figure 1).

As the main challenge for leveraging the generative model is to
generate a new dataset, with both new training vectors and their
labels. Otherwise, without their class labels, the new dataset may
only be applied in unsupervised learning tasks. To address this chal-
lenge, we propose to build a multi-modal variational autoencoder
motivated by Gaussian Mixture Models [7]. Conceptually, each
mode of VAE is used to capture the distribution of the data for each
class. Thus, the entire dataset is modeled by the mixture of these
modes. Traditionally, Linear Discriminant Analysis (LDA), Bayes
Net, and mixture of Gaussians also utilize this type of generative
models; henceforth this multi-modal model is shown to be effective
for classification.
Figure 1: Overview of proposed differentially private data sharing approaches. Differentially private data sharing of private data $X$ is achieved by 1) sharing a representation learning model (DP-AuGM) trained on the private data $X$, and 2) by generating new surrogate data $X'$ via a generative model (DP-VaeGM). After publishing $X'$, different learning models can be trained on $X'$ to protect privacy of $X$ while achieving high learning accuracy (data utility).

**Algorithm 1: DP-AuGM**

More specifically, DP-VaeGM proceeds as below and the algorithm is outlined in Algorithm 2:

- Firstly, it initializes with $n$ variational autoencoders (VAEs), where $n$ is the number of the classes for the specific data. Each model $M_i$ is responsible for generating the data of a specific class $1 \leq i \leq n$. We empirically observe that training $n$ generative models results in higher utility than training a single model; this is because a single model would need to capture the class label latent variables following a Gaussian distribution. Using $n$ separate models can also generate a balanced dataset even if the original data are imbalanced.

- Secondly, it uses a differentially private training algorithm (such as DP-DL) to train each generative model $M_i$.

- Finally, it samples data from Gaussian distribution $N(0, 1)$ for the sampling layer of each variational autoencoder. It returns the entire generated data $X'$ by taking the union of generated data from each generative model $M_i$.

**DP Analysis for DP-VaeGM.** We have adopted the algorithm developed by Abadi et al. [4] to train each VAE. Thus each training algorithm is $O(qe \sqrt{T})$, $\delta$-differentially private. Next we prove that each variational autoencoder (VAE) is a differentially private generative model (see Theorem 1) and the entire DP-VaeGM is also $O(qe \sqrt{T})$, $\delta$-differentially private (see Theorem 2). Formally, to show proofs, we let $X$ be the shared data, $\Theta$ be model parameters, and $X'$ be the generated data (the output of a single VAE).

**Theorem 1.** Let $\mathcal{T} : X \rightarrow \Theta$ be a VAE training algorithm that is $(\epsilon, \delta)$-differentially private based on [4]. Let $f : \Theta \rightarrow X'$ be a mapping that maps model parameters to output, with Gaussian noise generated from a sampling layer of VAE as input. Then $f \circ \mathcal{T} : X \rightarrow X'$ is $(\epsilon, \delta)$-differentially private.

**Proof.** The proof is immediate by applying the post processing property of differential privacy [13].

**Theorem 2.** Let a generative model (VAE) of class $i M_i : X_i \rightarrow X'_i$ be $(\epsilon, \delta)$-differentially private. Then $\mathcal{G}_n : X \rightarrow \Pi_{i=1}^n X'_i$ is defined to be $\mathcal{G}_n = \bigcup_{i=1}^n M_i$. Then $\mathcal{G}_n$ is $(\epsilon, \delta)$-differentially private, for any integer $n$.

**Proof.** Given two adjacent datasets $X_1$ and $X_2 = X_1 \cup \{b\}$, without loss of generality, we assume $b$ belongs to class $k (1 \leq k \leq n)$. Fix any subset of events $S \subseteq \Pi_{i=1}^n X'_i$. Since the $n$ generative models are pairwise independent, we obtain $\Pr[\mathcal{G}_n(X_i) \in S] = \Pi_{i=1}^n \Pr[M_i(x_i) \in S]$, where $x_i \subseteq X_i = \bigcup_{i=1}^n x_i$ denotes
the training data of $X_i$ for the $i$th generative model. Similarly, $\Pr[G_n(X_i) \in S] = \Pi^n_{i=1} \Pr[M_i(x_i^k) \in S]$. Since $X_i$ and $X_j$ only differ in $i$, we have $x_i^k = x_j^k$ and $\Pr[M_i(x_i^k) \in S] = \Pr[M_j(x_j^k) \in S]$, for any $i \neq k$. Since $M_k$ is $(\epsilon, \delta)$-differentially private, then we have $\Pr[M_k(x_i^k) \in S] \leq e^\epsilon \Pr[M_k(x_j^k) \in S] + \delta$. Therefore, we obtain $\Pr[G_n(X_i) \in S] = \Pi^n_{i=1} \Pr[M_i(x_i^k) \in S] = \Pr[M_1(x_1^k) \in S] \times \cdots \times \Pr[M_k(x_k^k) \in S] \times \cdots \times \Pr[M_n(x_n^k) \in S] \leq e^\epsilon \Pi^n_{i=1} \Pr[M_i(x_i^k) \in S] + \delta = e^\epsilon \Pr[G_n(X_i) \in S] + \delta$. The inequality derives from the fact that any probability is no greater than 1. Hence, $G_n$ is $(\epsilon, \delta)$-differentially private, for any $n$.

4 EVALUATION

4.1 Datasets

**MNIST.** MNIST [22] is the benchmark dataset containing handwritten digits from 0 to 9, comprised of 60,000 training and 10,000 test examples. Each handwritten grayscale image of digits is centered in a $28 \times 28$ or $32 \times 32$ image. To be consistent with [20], we choose to use the $32 \times 32$ version of MNIST dataset when evaluating our generative models against the GAN-based attack.

**Adult Census Data.** The Adult Census Dataset [23] includes 48,843 records with 14 sensitive attributes, including gender, education level, marital status, and occupation. This dataset is commonly used to predict whether an individual makes over 50K dollars in a year. 32,561 records serve as a training set and 16,282 records are used for testing.

**Hospital Data.** This dataset is based on the Public Use Data File released by the Texas Department of State Health Services in 2010Q1 [2]. Within the data, there are personal sensitive information, such as user permission request. We randomly choose 36,000 instances as testing data and the rest serves as the training data.

**Malware Data.** This dataset is gathered by using 10% training data and combining it with DP-AuGM prior to releasing information about the shared data. DP-AuGM trained over the shared data can better capture the inner representations of the dataset, which boosts the following learning accuracy of machine learning models.

4.2 Evaluation of DP-AuGM

In this subsection, we show how DP-AuGM performs in terms of utility under the various differential privacy budget on four datasets. To evaluate performance, for all the four datasets, we assume 90% of the training data is used as shared data while the remaining 10% still serves as the training data. To demonstrate how DP-AuGM helps boost the learning efficacy, we compare the learning efficacy between: when only using 10% training data and when combining it with DP-AuGM for better data representations.

**Effect of the Privacy Budget.** To evaluate the effects of the privacy budget (i.e., $\epsilon$ and $\delta$) on prediction accuracy for machine learning models, we vary $(\epsilon, \delta)$ to test learning efficacy (i.e., the utility metric) on different datasets. The results are shown in Figures 2(a)-(d). In these figures, each curve corresponds to the best accuracy achieved given a fixed $\delta$, as $\epsilon$ varies between 0.2 and 8. In addition, we also show the baseline accuracy (i.e., without DP-AuGM) on each dataset for the comparison. From Figure 2, we can see that the prediction accuracy decreases as the noise level increases ($\epsilon$ decreases), while we see DP-AuGM can still achieve comparable utility with the baseline even when $\epsilon$ is tight (i.e., around 1). When $\epsilon = 8$, for all the datasets, the accuracy lags behind the baseline within 3%. This demonstrates that data generated by DP-AuGM can preserve high data utility for subsequent learning tasks.

**Efficacy of DP-AuGM.** We further examine how DP-AuGM helps boost the learning efficacy. We compare the learning accuracy between using 10% training data and combining it with DP-AuGM for getting better data representations. For DP-AuGM, we set the private budget $\epsilon$ and $\delta$ to be 1 and $10^{-5}$, respectively. We do the comparisons on all the datasets and the result is presented in Table 1. As we can see from Table 1, after using DP-AuGM, the learning accuracy increases by at least 6% on all the datasets and by 34% on Malware Data dataset. This demonstrates the significance of using DP-AuGM prior to releasing information about the shared data. DP-AuGM trained over the shared data can better capture the inner representations of the dataset, which boosts the following learning accuracy of machine learning models.

<table>
<thead>
<tr>
<th>Datasets</th>
<th>With DP-AuGM</th>
<th>Without DP-AuGM</th>
</tr>
</thead>
<tbody>
<tr>
<td>MNIST</td>
<td>0.95</td>
<td>0.89</td>
</tr>
<tr>
<td>Adult Census Data</td>
<td>0.78</td>
<td>0.64</td>
</tr>
<tr>
<td>Hospital Data</td>
<td>0.56</td>
<td>0.42</td>
</tr>
<tr>
<td>Malware Data</td>
<td>0.96</td>
<td>0.62</td>
</tr>
</tbody>
</table>

4.3 Evaluation of DP-VaeGM

In this subsection, we empirically evaluate utility performance of our proposed data generative model DP-VaeGM. As VAE is typically used to generate high quality images, now we only evaluate DP-VaeGM on the MNIST image dataset.

**Table 1: Comparisons of training accuracy between using only public data for training and using both DP-AuGM and public data**
Table 2: Comparisons between DP-AuGM and sPATE on MNIST

<table>
<thead>
<tr>
<th>Models</th>
<th>Privacy budget $\epsilon$</th>
<th>Privacy budget $\delta$</th>
<th>Accuracy</th>
<th>Baseline</th>
</tr>
</thead>
<tbody>
<tr>
<td>sPATE [27]</td>
<td>1.97</td>
<td>$10^{-5}$</td>
<td>0.985</td>
<td>0.992</td>
</tr>
<tr>
<td>DP-AuGM</td>
<td>1.97</td>
<td>$10^{-5}$</td>
<td>0.987</td>
<td>0.992</td>
</tr>
</tbody>
</table>

Effect of the Privacy Budget. We vary the privacy budget to test DP-VaeGM on MNIST dataset. The result is shown in Figure 3, where each curve corresponds to the best accuracy given $\delta$, and $\epsilon$ varies between 0.2 and 8. We plot the baseline accuracy (i.e., without DP-VaeGM) using the red line. From this figure, we can see that DP-VaeGM can achieve comparable utility w.r.t. the baseline. For instance, when $\epsilon$ is greater than 1, the accuracy is always higher than 92%. When $\epsilon$ is 8 and $\delta$ is $10^{-2}$, the accuracy is over 97% which is lower than the baseline by 2%. Thus, we can see that DP-VaeGM has the potential to generate data with high training utility while providing privacy guarantees for private data.

Table 3: Comparisons between DP-VaeGM and sPATE on MNIST

<table>
<thead>
<tr>
<th>Models</th>
<th>Privacy budget $\epsilon$</th>
<th>Privacy budget $\delta$</th>
<th>Accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>sPATE [27]</td>
<td>1.97</td>
<td>$10^{-5}$</td>
<td>0.985</td>
</tr>
<tr>
<td>DP-VaeGM</td>
<td>1.97</td>
<td>$10^{-5}$</td>
<td>0.968</td>
</tr>
</tbody>
</table>

as specified in [27]. As sPATE requires the presence of public data, we split the test data into two parts in the same way as specified by [27]. Considering DP-VaeGM does not need public data, the private data is discarded for DP-VaeGM. In addition, the privacy budget $\epsilon$ and $\delta$ is set to be 1.97 and $10^{-3}$, respectively. From Table 3, we can see that DP-VaeGM falls behind sPATE by approximately 2%. This is because that sPATE trains the model using both public and private data while DP-VaeGM is only trained with private data.

5 CONCLUSION

We have designed, implemented, and evaluated two approaches of differentially private data sharing via a differentially private autoencoder-based generative model (DP-AuGM) and a differentially private variational autoencoder-based generative model (DP-VaeGM), respectively. We show that both approaches can provide provable privacy guarantees and retain high data utility for machine learning tasks. We hope that our work will help pave the way toward designing more effective differentially private data sharing approaches in the dynamic digital world.

ACKNOWLEDGMENTS

This work was supported in part by National Science Foundation of China, under Grant No. 61672350.

REFERENCES

Table 4: Model structures of DP-AuGM over different datasets

<table>
<thead>
<tr>
<th>MNIST</th>
<th>Adult Census Data</th>
<th>Texas Hospital Stays Data</th>
<th>Malware Data</th>
</tr>
</thead>
<tbody>
<tr>
<td>FC(400)+Sigmoid</td>
<td>FC(4)+Sigmoid</td>
<td>FC(400)+Sigmoid</td>
<td>FC(50)+Sigmoid</td>
</tr>
<tr>
<td>FC(256)+Sigmoid</td>
<td>FC(100)+Sigmoid</td>
<td>FC(77)+Sigmoid</td>
<td>FC(142)+Sigmoid</td>
</tr>
<tr>
<td>FC(400)+Sigmoid</td>
<td>FC(784)+Sigmoid</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 5: Model structures of DP-VaeGM over MNIST

<table>
<thead>
<tr>
<th>MNIST</th>
</tr>
</thead>
<tbody>
<tr>
<td>FC(500)+Sigmoid</td>
</tr>
<tr>
<td>FC(500)+Sigmoid</td>
</tr>
<tr>
<td>FC(20)+Sigmoid</td>
</tr>
<tr>
<td>Sampling Vector(20)</td>
</tr>
<tr>
<td>FC(500)+Sigmoid</td>
</tr>
<tr>
<td>FC(784)+Sigmoid</td>
</tr>
</tbody>
</table>

Table 6: Structures of machine learning models over different datasets with DP-AuGM

<table>
<thead>
<tr>
<th>MNIST</th>
<th>Adult Census Data</th>
<th>Texas Hospital Stays Data</th>
<th>Malware Data</th>
</tr>
</thead>
<tbody>
<tr>
<td>Conv(5x5,3,32)+Relu MaxPooling(2x2,2,2)</td>
<td>Conv(5x5,3,32)+Relu MaxPooling(2x2,2,2)</td>
<td>Reshape(4x4x32)</td>
<td>FC(10)</td>
</tr>
<tr>
<td>Conv(5x5,3,32)+Relu MaxPooling(2x2,2,2)</td>
<td>Conv(5x5,3,32)+Relu MaxPooling(2x2,2,2)</td>
<td>Reshape(4x4x32)</td>
<td>FC(10)</td>
</tr>
<tr>
<td>Conv(5x5,3,32)+Relu MaxPooling(2x2,2,2)</td>
<td>Conv(5x5,3,32)+Relu MaxPooling(2x2,2,2)</td>
<td>Reshape(4x4x32)</td>
<td>FC(10)</td>
</tr>
</tbody>
</table>

Table 7: Structures of machine learning models over different datasets with DP-VaeGM

<table>
<thead>
<tr>
<th>MNIST</th>
</tr>
</thead>
<tbody>
<tr>
<td>Conv(5x5,3,32)+Relu MaxPooling(2x2,2,2)</td>
</tr>
<tr>
<td>Conv(5x5,3,32)+Relu MaxPooling(2x2,2,2)</td>
</tr>
<tr>
<td>Conv(5x5,3,32)+Relu MaxPooling(2x2,2,2)</td>
</tr>
</tbody>
</table>