



Effect of Different Data Augmentation Strategies on Performance In Federated Learning Systems

Lohithsai Yadala Chanchu
Supervisors: David Tax, Swier Garst

¹**EEMCS, Delft University of Technology, The Netherlands**

A Thesis Submitted to EEMCS Faculty Delft University of Technology,
In Partial Fulfilment of the Requirements
For the Bachelor of Computer Science and Engineering
June 23, 2024

Name of the student: Lohithsai Yadala Chanchu
Final project course: CSE3000 Research Project
Thesis committee: David Tax, Swier Garst, Alex Voulimeneas

An electronic version of this thesis is available at <http://repository.tudelft.nl/>.

Abstract

Federated learning (FL) enables privacy-preserving collaboration among numerous clients for training machine learning models. In FL, a server coordinates model aggregation while preserving data privacy. However, non-identically and independently distributed (non-IID) local data label distributions degrade the performance of the global model. This paper investigates the impact of synthetic data on mitigating non-IID data distributions in federated learning. We explore data-based augmentation techniques, including uniform and minority imputation, utilizing conditional variational autoencoders (CVAEs) to generate synthetic data. Additionally, we examine a framework-based approach where a pre-trained model, centrally trained on synthetic data, is distributed to clients for fine-tuning on their original datasets. Our results, which use the binarized MNIST dataset, demonstrate a quality gap between synthetic and original datasets, leading to diminished classification performance when trained on only synthetic data. Integrating both original and synthetic data improves performance on heavily imbalanced label distributions. At the same time, uniform imputation experiments reveal that optimal imputation must strike a balance, with performance degradation being noticeable when datasets consist of more than 45% synthetic images. Synthetic imputation did not suffer from degradation in the explored range of imputation amounts and achieved an average F1 score improvement of 0.015 over uniform imputation.

1 Introduction to Federated Learning

In recent years, increasingly larger and more complex deep learning models are being developed that achieve accuracies in tasks that were not possible before. These models need massive amounts of high-quality data to be trained effectively, which conflicts with constraints related to data privacy regulations and restrictions. Due to privacy concerns and data regulations such as GDPR, the handling of data has become more fragmented [27]. This has led to the emergence of distributed databases comprised of multiple data silos, spread across mobile devices, institutions, and countries which cannot be freely transferred. This is especially a major hindrance to applying deep learning techniques to privacy-focused industries, such as the medical and financial fields. To meet these privacy constraints while still producing competitive models, Google introduced the Federated Learning (FL) algorithm [19]. In FL systems, decentralized clients update a global model by training on local data as seen in Figure 1.

The central server coordinates the training process, collecting the locally trained models, and aggregating them to form an updated global model (which is then sent to the clients to train on further iterations). This ensures local models can get the benefit of a bigger dataset while maintaining the privacy of their data. The original FL aggregating algorithm, FedAvg [19] updates the global model by averaging

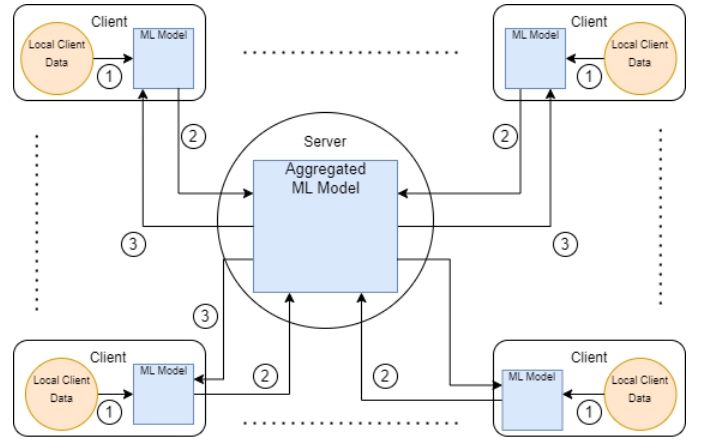


Figure 1: Overview of the basic federated learning algorithm: Local ML models are trained with local data (1). The local models are aggregated at the server (2). This aggregated model is sent back to clients to be trained further on local data (3).

the clients' updated parameters weighted by the size of their respective datasets. However, the FL algorithm as originally proposed leads to a substantial worsening in accuracy when each client's data is heterogeneous. For instance, local client datasets that are not identically and independently distributed (IID) can significantly hamper global model performance due to the discrepancy between the global model and local model parameters during training [30] [29]. One way data can be non-IID is if each client has major differences in the distribution of labels of its dataset. This problem has been tackled in multiple ways which can be broadly categorized into data-based, model-based, algorithm-based, and framework-based approaches [18]. Data sharing [29] is an example of a data-based approach, where the data distribution is transformed in some way. However, the downside is the major loss of privacy. Data augmentation approaches aim to address this pitfall using generative methods rather than original local data [5]. Data points, which are synthesized from generative models, are then imputed, or inserted, into the original dataset, to make client datasets more IID. Framework-based solutions, such as similarity clustering, knowledge distillation, and transfer learning aim to solve the non-IID problem by modifying the overall FL framework instead of the data distribution itself.

In this paper, the effectiveness of different techniques to augment data in a federated setting is explored. We aim to investigate to what extent different data augmentation methods such as uniform and minority imputation affect FL performance under a variety of imbalanced dataset conditions. We also focus on local client data augmentation using data and framework-based solutions by finetuning a centrally trained classifier trained with synthetic images produced by a variational autoencoder (VAE) trained in a federated manner.

Section 2 introduces VAEs and presents relevant work which discusses solutions that have been proposed to mitigate the non-IID problem. Section 3 elaborates on the different imputation techniques that are investigated. It also outlines the steps that were used to conduct the analysis, which

is motivated by the presented preliminary findings. The experimental setup and the corresponding results are described in Section 4. These results are further discussed in Section 5. Section 6 concludes our findings, mentions the limitations of the study and presents opportunities for future work.

2 Background

2.1 VAEs

VAEs and autoencoders share a similar structure, as illustrated in Figure 2. They both comprise an encoder, which maps the input data x to a point in a latent space. This latent point is then used to train a decoder to return an output $d(z)$ that matches the input as closely as possible.



Figure 2: Overview of the variational autoencoder architecture

However, VAEs differ in that they encode inputs as distributions, instead of points, by returning parameters to a parameterized distribution, such as the normal distribution. To ensure continuity (where nearby points decode into similar outputs) and completeness (where sampled points decode into "meaningful" outputs), we must regularize both the covariance matrix and the means of the distributions returned by the encoder, to encourage the returned distributions to be close to each other. This involves enforcing that the returned distributions closely resemble the standard normal distribution, thereby ensuring that the individual class distributions are closely aligned. This is done by a loss function that balances the reconstruction term with the Kullback-Leibler divergence, which measures the "distance" the current distribution is to a standard normal distribution [15].

Conditional VAEs are a kind of VAE which, given a label, generate an output corresponding to the label [25]. This is different to traditional VAEs, where a random point in the latent space is given as the input to be decoded, which generates outputs with random labels.

2.2 Related Generative Solutions to the Non-IID Problem

Non-IID problems in federated learning have been tackled in multiple ways. An example of a framework-based approach is outlined in [22], which proposes a cluster-based approach to federatively train an ensemble of models, each of which is aggregated from clients with similar labels. This helps mitigate model shifts since only similar clients will be aggregated together. However, these techniques usually come with a performance burden on the central server. For example, with [22], the server needs to store multiple models and figure out which client models should aggregate with each other, which is computationally expensive.

Other classes of solutions, such as data-based augmentation techniques, have also been investigated using a variety of generative models. This class of solutions aims to make the local client data more IID, which in turn reduces model

drift, by imputing synthetic data such that the label distribution becomes more similar between clients. Generative Adversarial Networks (GANs) have become a popular choice for federated data augmentation [17], where synthetic data is predominantly generated locally at each client. Related work has focused on exploring various GAN-based frameworks by adjusting the locations for training the generator and discriminator models [2] [6] [11]. Nevertheless, training GANs in a federated way comes with its own challenges. GANs typically require large amounts of data to perform effectively. Even with adequate data, instability issues frequently arise, even during centralized training [9]. Other generative models, such as VAEs have also been explored. [4] trains only a global decoder and shares the latent means of some noisy images from each client's respective datasets.

On the other hand, data-sharing schemes have been proposed, where the actual data itself is being sent to other clients, instead of the generative model parameters. For example, [13] shares a limited set of real local data. However, this violates the privacy-preserving principles of FL. Another data-sharing method is proposed in [23], where synthetic data is transferred. This synthetic data is generated by performing XOR decoding on its samples, which is then shared between clients. This overcomes the previous limitation of sharing real data. Nevertheless, sharing data, especially in large-scale systems with thousands of clients, causes a huge communication overhead, which slows down the process of federated learning and puts a heavy burden on the communication infrastructure used in these networks.

In this paper, different data-based augmentation techniques, such as uniform and minority imputation, and framework-based techniques involving fine-tuning are investigated.

3 Methodology

We conducted our experiments on the MNIST [7] dataset with varying degrees of imbalances. MNIST is a dataset of 28x28 black-and-white images of handwritten digits. Similar to previous studies, we created partitions of different label skews by using the Dirichlet distribution [28] by varying the β parameters. The smaller the β parameter, the greater the imbalance tends to be, meaning that the label distribution on clients will become increasingly skewed as seen in Figure 3. We used $\beta = [0.1, 0.3, 0.5, 10.0, 100.0]$ to compare the performance on datasets with a range of imbalances among client label distributions.

The use of a conditional VAE (CVAE), which generates the appropriate data points given a label, is needed here as preliminary experiments with vanilla VAEs showed that the distribution of the generated labels under federated learning became increasingly skewed as the data imbalance worsened. As seen in Figure 4, even though the MNIST dataset has a near-uniform label distribution, the corresponding generated label distribution is heavily skewed. CVAEs address this issue by separating the process of image generation from the selection of the label to be generated. The CVAE architecture we used is taken from [25].

The training was split into two stages. First, the CVAE was

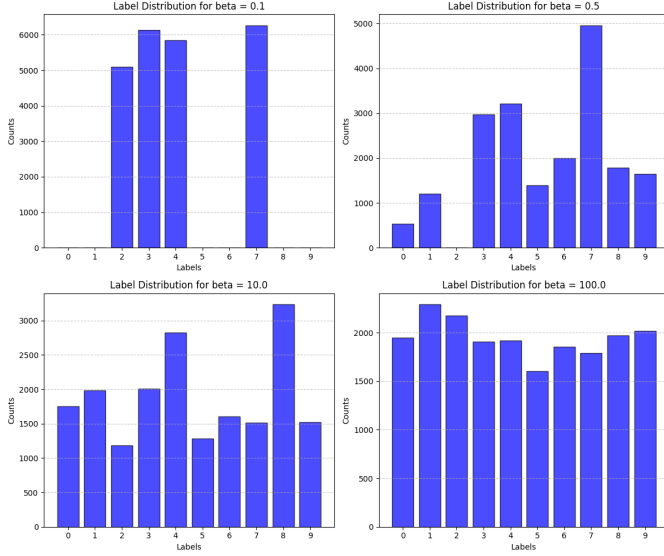


Figure 3: Label distribution for select clients for various beta values

trained in a federated manner. This trained CVAE was used to augment the dataset in various ways. The augmented dataset is then used by a classifier, which is used to classify the labels of the generated or original images of digits.

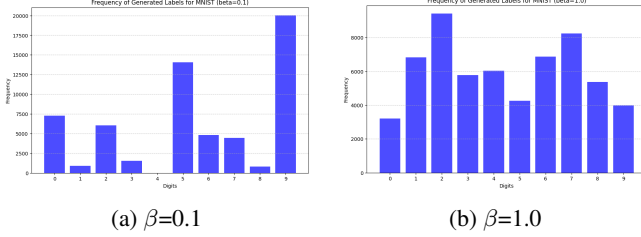


Figure 4: Influence of data imbalance on VAE-generated label distribution

3.1 Imputation Investigation

The first two methods are variations of the Synthetic Minority Reconstruction Technique [24], which involves locally imputing k synthetic data points using a federated CVAE and also training a classifier in a federated manner as seen in Figure 5, where a CVAE trained in a federated manner is used to impute k data points into the client datasets. The effect k has on the client test metrics was investigated.

Two imputation techniques, namely uniform and minority imputation, were investigated. Uniform imputation imputes all the labels uniformly, no matter what the initial local client label distribution is. Minority imputation takes the prior label distribution into account, imputing more synthetic data points with the minority labels, as shown in Figure 6. The number of imputed data points generated per label is specified by Algorithm 1, where the bigger the difference between the most common label frequency and the frequency of a certain label, the more that certain label is imputed. This algorithm assumes a global, uniform label distribution. If the global

distribution is not uniform, then an additional step to collect and aggregate label counts from clients is needed. This information must then be used to adjust the imputation amount such that each local distribution is imputed so that it matches this global label distribution.

These two imputation approaches were investigated to determine whether the potential performance improvement is due to simply adding new data indiscriminately or if it is the result of making client datasets more similar, a task that minority imputation accomplishes more effectively.

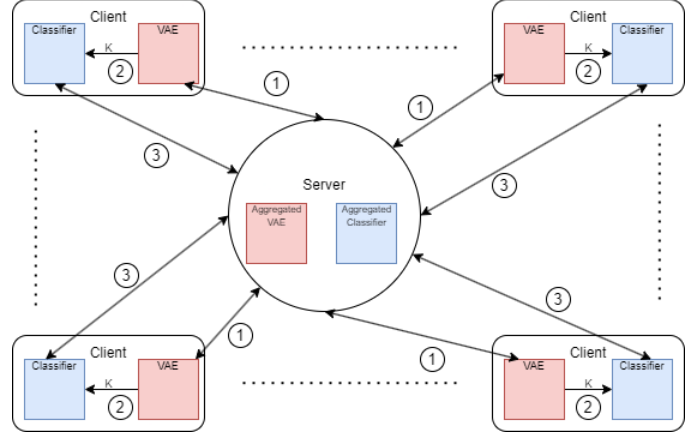


Figure 5: Overview of steps in data-based augmentation experiments: The CVAE is trained in a federated manner (1). The trained CVAE is used to augment k data points into the local dataset (2). The classifier is trained on the augmented local dataset in a federated manner (3).

3.2 Finetuning Investigation

The second method, as shown in Figure 7, looked into the performance of the federated system if the classifier was trained centrally with the generated data. The VAE is still trained in a federated manner. This pre-trained classifier would then be fine-tuned locally at each client using original data. We simulated data points with the same label distribution as the global MNIST dataset with the CVAE to get 70k images, consisting of 60k training and 10k testing images, similar to the original MNIST dataset. We then test the performance of the classifier on the clients each using their respective original (non-imputed) local test datasets for each method. We compare these methods with the baseline of no data augmentation with varying label imbalances.

This method allows us to also assess the quality of the generated data for the classification task from the intermediate results of the pre-trained models. This approach also allows us to compare the effectiveness of transfer learning in federated learning to the data-based imputation methods previously discussed in Section 3.1.

3.3 Experiment Setup

We extended the implementation of the vanilla FL framework used in [19], which was written in Pytorch. Our code can

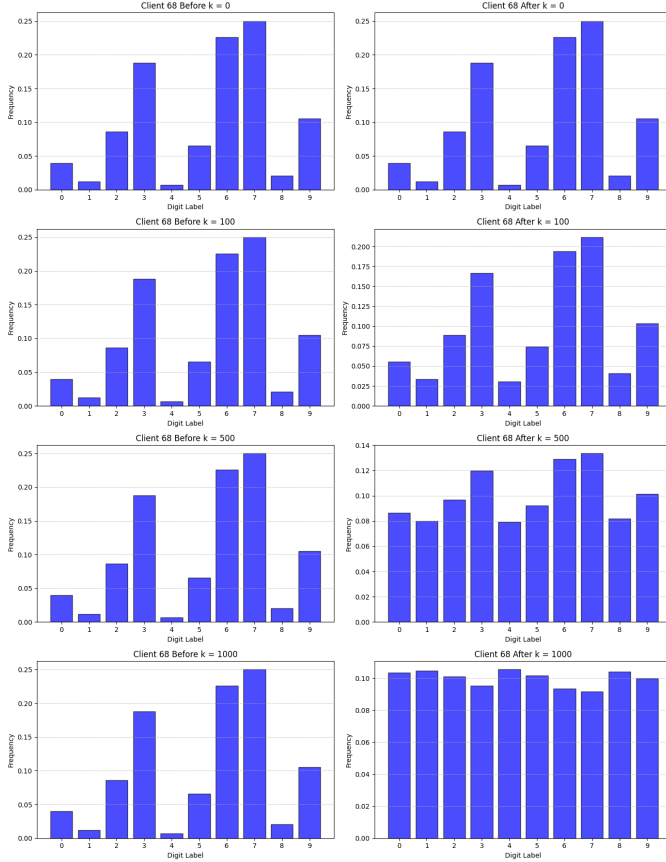


Figure 6: Local label distribution from an example client with $\beta=0.5$ before and after doing minority imputation with various imputation amounts

be found on Github¹. All the models were trained using the Adam [14] optimizer. We simulated the federated learning process using 100 clients with all of them participating every round. The binary cross entropy loss was used to train the classifier. The aggregation algorithm is FedAvg. Each experiment was run 5 times to ensure reliable results and to determine the variability of the results. At the beginning of each run, a CVAE was trained federatively on data split according to $\beta = [0.1, 0.3, 0.5, 10.0, 100.0]$. All the errors represented in the Figures are ± 1 standard deviation away from the mean of the trials.

We use ExquisiteNet [3] as our classifier model since it achieves a high classification accuracy and is complex enough for its performance to be noticeably degraded under federated conditions. To allow for increased performance of the generative models, we binarized the MNIST dataset, similar to [21].

Since local clients will be dealing with datasets with an imbalanced label distribution, we must look at the F1 score in addition to the accuracy of the classifier such that we minimize the metric bias caused by the majority class [1].

¹<https://github.com/Lyadalachanchu/Effect-of-Different-Data-Augmentation-Strategies-on-Performance-In-Federated-Learning-Systems>

Algorithm 1 CVAE-based Minority Class Imputation

```

1: Input:  $k, trained\_cvae, initial\_ds, label\_distribution$ 

2: Output:  $combined\_ds$ 
3:  $deficits \leftarrow \emptyset$ 
4: for each  $label, count$  in  $label\_distribution$  do
5:    $deficits[label] \leftarrow \max(label\_distribution) - count$ 
6: end for
7:  $inverse\_counts \leftarrow k \times \text{Normalize}(deficits)$ 
8: for each  $label$  in  $label\_distribution$  do
9:    $num\_samples \leftarrow \text{Round}(inverse\_counts[label])$ 
10:  if  $num\_samples > 0$  then
11:     $generated\_images \leftarrow \text{GenerateData}(trained\_cvae, num\_samples, label)$ 

12:   $generated\_ds \leftarrow \emptyset$ 
13:  for each  $image$  in  $generated\_images$  do
14:     $generated\_ds \leftarrow generated\_ds \cup (image, label)$ 
15:  end for
16: end if
17: end for
18: return  $\text{CombineDatasets}(initial\_ds, generated\_ds)$ 

```

4 Results

4.1 Imputation Experiments

$K = [0, 100, 500, 1000]$ images were generated using the CVAE that was trained on the respective β parameter for each client. With an average of 600 original images on each client, synthetic images make up 0%, 14.2%, 45%, and 62.5% of the training data for their respective imputation amounts. The number of local epochs and global communication rounds was set to 1 and 15, respectively, as increasing them did not significantly enhance the performance of any augmentation method. The classifier was trained with a learning rate of 0.01 and a weight decay of 0.0001 with a batch size of 32. These settings were used for both uniform and minority imputation.

Uniform Imputation

The mean test losses and their variances do not seem to be affected by the number of imputed data points, as seen in figure 8. However, this does not seem to be the case with the final test macro F1 scores shown in Figure 9. The F1 score variance is lower for higher values of β , however, their variances do not seem to be affected by k , similar to the test losses. F1 performance seems to be affected the most when the data is the most imbalanced (for smaller values of β). The macro F1 performance saw a noticeable increase for the two most imbalanced data splits $\beta = [0.1, 0.3]$ between no imputation and 100 imputed images. However, it's also apparent that too many imputed images seem to degrade performance as can be seen with $k=1000$ in Figure 10.

Minority Imputation

As seen in Figure 12, the average F1 score across all β s is higher when using minority imputation compared to when using uniform imputation. Unlike with uniform imputation, there seems to be a continuous performance improvement, at

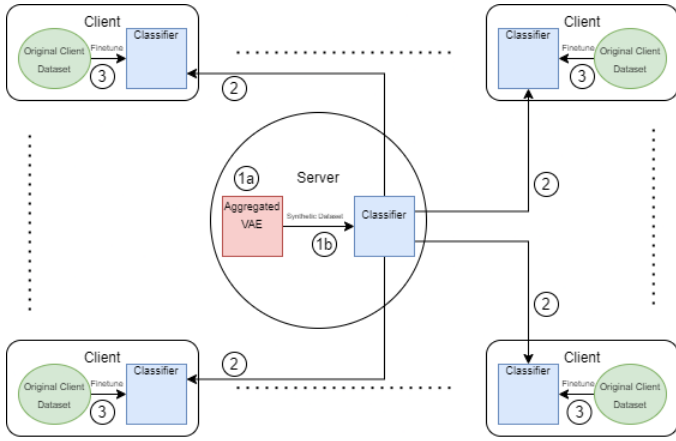


Figure 7: Overview of steps in finetuning experiments. The aggregated CVAE is trained in a federated manner (1a). The CVAE generates a synthetic dataset which the central classifier uses to train itself locally on the server (1b). This pre-trained classifier is sent to each client (2), where it is further trained with original data from the local dataset (3).

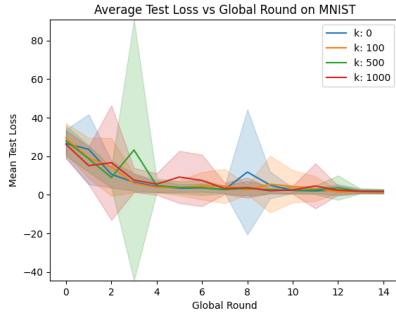


Figure 8: Test Losses for Each Global Round for Different Imputation Amounts

least for the range for the explored values of k . Although this performance improvement is diminishing the higher the k . This can be seen in Figure 12 where the F1-score improves more between $k=100$ and $k=500$ than between $k=500$ and $k=1000$. Synthetic imputation demonstrates steady performance improvement, even with more balanced client datasets, as shown in Figure 11. There is a roughly 0.1 increase in the F1-score between $k=0$ and $k=1000$ when $\beta=100.0$. In contrast, uniform imputation with higher β s results in performance stagnation with less drastic improvements between $k=0$ and $k=1000$ for β s = [10.0, 100.0].

4.2 Finetuning Experiments

For each of the 5 runs, a synthetic dataset was created for each beta parameter with their respective CVAEs. 70,000 synthetic MNIST images were generated with the same label distribution as the original MNIST dataset using their respective CVAEs. The five classifiers were trained in a centralized manner on their respective synthetic dataset with 15 epochs, a learning rate of 0.01 and a weight decay of 0.0001 with a batch size of 32. Each classifier was then sent to each client,

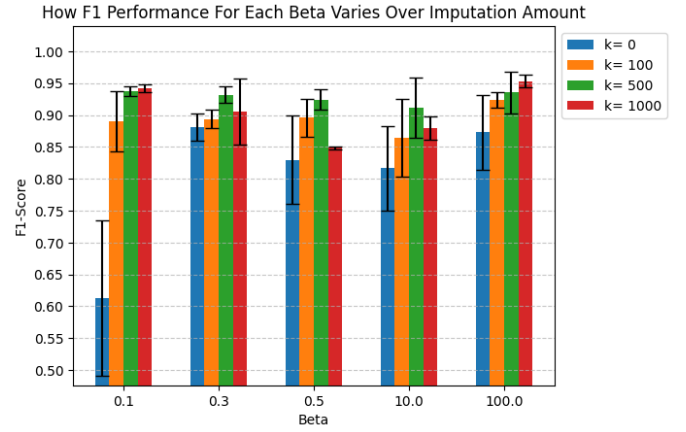


Figure 9: Test macro-F1 scores for different imputation amounts with uniform imputation

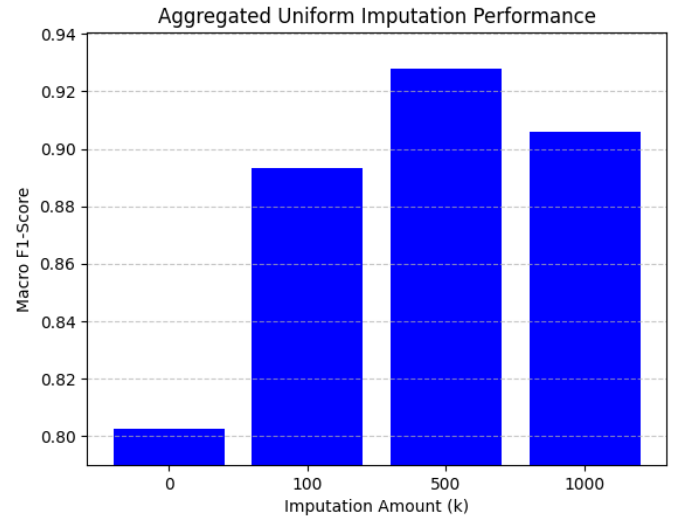


Figure 10: Aggregated test macro-F1 scores for different imputation amounts with uniform imputation

which then locally trained the classifier with local MNIST data (split according to the respective beta parameter of the data the classifier was trained on) for 8 local epochs with all other hyperparameters remaining constant.

A large discrepancy between the training (performed on synthetic MNIST data) and test (performed on original MNIST) loss is apparent as the test losses are roughly a magnitude of order higher than the train losses as shown by Table 1. This difference is reflected in the test accuracies shown in Figure 13. The accuracies for the classifier trained on synthetic data all start significantly higher than the accuracies for the classifier trained on the original dataset as seen in Figures 13a and 13b respectively.

Further exploration was performed to investigate the cause for this discrepancy in the performances between the synthetic and original data. Looking at the output probabilities in Figure 14, the model was more uncertain with real data,

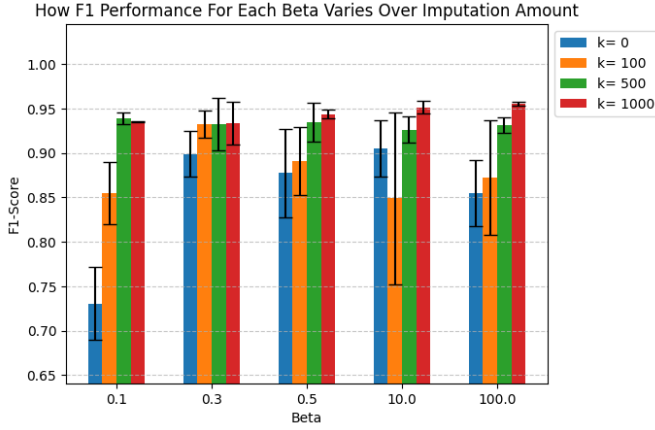


Figure 11: Test macro-F1 scores for different imputation amounts with minority imputation

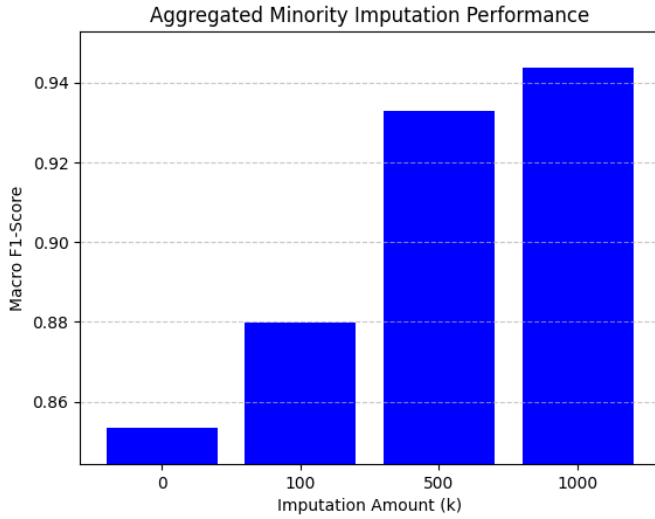


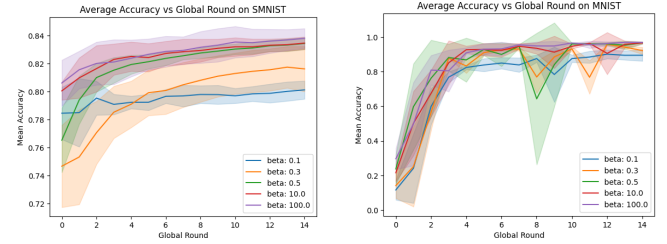
Figure 12: Aggregated test macro-F1 scores for different imputation amounts with minority imputation

which might partly explain the discrepancy between the train and test losses. However, the accuracies also seem to be significantly lower when trained on synthetic data, while the model is relatively confident most of the time, which means that when the model is wrong, it heavily contributes to the cross entropy loss since it is so confident.

This discrepancy between the synthetic and original data means that the pre-trained classifier performs significantly worse than the finetuned and the original federated strategies as seen in Figure 15. However, after finetuning the pre-trained classifier, we see better performance than the vanilla federated strategy for heavily imbalanced data while for more balanced data, they seem to be comparable. It also seems like finetuning reduces the variance of the performance.

Table 1: Final train and test losses for respective beta parameters with the model trained only on synthetic data and tested on original data

β	Train Loss	Test Loss
0.1	0.014241	1.374088
0.3	0.009557	1.865668
0.5	0.010085	1.226184
10.0	0.010298	1.359792
100.0	0.008356	1.546940



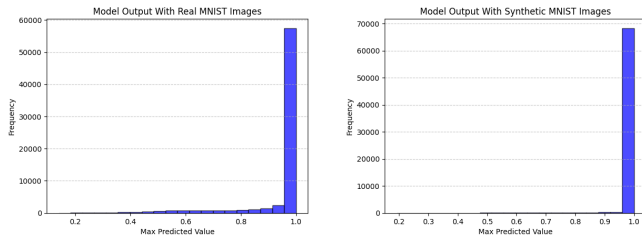
(a) Accuracies for classifier trained on synthetic MNIST data (b) Accuracies for classifier trained on original MNIST data

Figure 13: Test accuracies on original data during training over varying β parameters

5 Discussion

The results seem to generally indicate a difference in quality between the synthetic and the original MNIST dataset, causing a worse classification performance for synthetic data. Indeed this is a known problem [20], where performance on downstream tasks such as classification can be degraded by up to 20% with certain VAE models. This is apparent, especially in the results from the finetuning experiments, where Figure 13 shows significant accuracy degradation for all values of β when the classifier was trained on synthetic data. This can be seen also in Figure 15, where the pre-trained classifier (which was done only with synthetic data) achieves lower performance compared to the other strategies. However, it seems that when the classifier is trained with original data alongside synthetic data, performance is higher than with original data for only the heavily imbalanced data. This can be seen in Figure 15 since the finetuned classifier achieves a slightly higher F1 performance than the classifier trained only with original data for $\beta = [0.1, 0.3]$. This can also be supported by Figure 9 since the F1 performance of the classifier for data split according to $\beta = [0.1, 0.3]$ increases when 100 images are imputed compared to when none are imputed.

However, there seems to be a balance in the optimal amount of synthetic images to impute for uniform imputation. Performance stops improving and seems to be degraded after more than 500 images are imputed, especially for the more imbalanced data splits. Furthermore, it seems that uniform data imputation does not affect performance when the data is more balanced. This might be because synthetic data quality is lower than real data quality, and if the data is already balanced to achieve near-optimal performance, then adding sub-optimal data does not provide any benefit. This is in con-



(a) Predicted class probabilities of the classifier on original MNIST dataset (b) Predicted class probabilities of the classifier on synthetic MNIST dataset

Figure 14: Output probabilities of the classifier trained on synthetic data

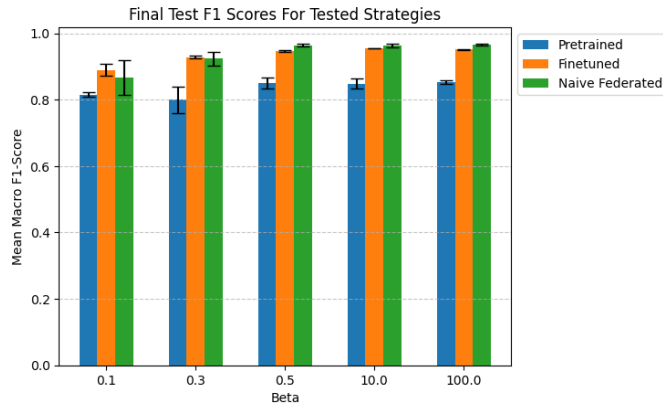


Figure 15: Test macro-F1 scores for pre-trained, finetuned, and vanilla federated strategies

trast to when the data imbalance is high, where imputing data allows the client label distributions to become more balanced, even though the imputed data is sub-optimal. This might also explain the presence of an optimal imputation amount since adding the first few images will have a greater effect in reducing data imbalance. However, once enough synthetic images are added, the benefit provided by the imbalance will not outweigh the degradation caused by the sub-optimal synthetic data.

However, with minority imputation, this degradation is not seen. This might be because each imputed data point is more likely to help make the datasets more IID than with uniform imputation resulting in an optimal imputation amount that is greater than the tested values of k . Unlike minority imputation, uniform imputation adds synthetic data points with the majority label, which doesn't provide much benefit to make the datasets more IID while diminishing the dataset quality. Minority imputation seems to be a better imputation method than uniform imputation since it achieves higher F1-scores across most values of k , except for $k=100$, as seen when comparing Figure 12 and Figure 10. On average, the F1 performance of minority imputation is 0.015 higher than with uniform imputation.

As seen in Figure 13a, the final accuracies of the classifier trained on synthetic data are lower than the final accuracies

when the classifier is trained on the original data for all β s. On the other hand, the start accuracies are also a lot higher when trained on synthetic MNIST images. This suggests that the generated digits do not look different enough from each other (unlike with real data) so the model learns only a few variations of each digit. This shows that the model produced digits with little variance. This could be fixed through the use of a beta VAE, which uses a modified loss function that allows the Kullback-Leibler divergence term to be weighted more heavily, allowing the generation of a wider variety of digit types.

Admittedly, the explored techniques also encounter some of the same pitfalls as related solutions, as discussed in Section 2. The finetuning approach increases server load. For this approach to work, the server will need to create and store a synthetic dataset and also train a model centrally, though this is a different type of computational complexity compared to the type faced by the framework-based solutions mentioned in 2.2. The explored imputation-based approaches overcome some of the privacy loss and communication burden limitations of the previous data-sharing methods, since the generative models themselves are being shared, not the images. This reduces the communication burden since sending the model is less data-intensive than sending the dataset itself. The privacy loss is minimized because we are not giving other clients access to the original data. However, no investigation into how much information the generated data leaks about the original dataset is done.

6 Conclusions and Future Work

Data heterogeneity is a significant challenge that impedes the effectiveness of practical federated learning systems, often leading to substantial performance degradation.

6.1 Main Contributions

This research addresses the data heterogeneity issue by investigating two data-based methods: synthetic uniform imputation and synthetic minority imputation. Additionally, a framework-based approach was explored, which involves locally fine-tuning a classifier that was initially trained centrally on synthetic data.

The key findings of the project are as follows:

- Combining original and synthetic data outperforms only original data in scenarios with extremely imbalanced label distributions and only synthetic data on all tested values of β .
- There appears to be an optimal amount of synthetic data imputation for uniform imputation, beyond which performance starts degrading.
- Minority imputation seems to lead to better performance than uniform imputation across most levels of imbalances, with the most improvement for the most imbalanced data.
- It was reaffirmed that using solely synthetic data for classifier training leads to significantly worse performance in downstream tasks.

These results highlight the potential of synthetic data augmentation to mitigate the effects of data heterogeneity in federated learning, offering insights into optimal strategies for improving model performance.

6.2 Limitations and Future Work

Our study presents several limitations that should be addressed in future research. Firstly, the experiments were conducted exclusively on the MNIST dataset. This narrow focus limits the generalizability of the results to other datasets, as different datasets may exhibit diverse characteristics and complexities.

Secondly, we did not utilize a β -VAE to control or increase the variance of the types of digits generated. Implementing a β -VAE could provide more diverse generative outputs, which may enhance the robustness and performance of federated learning models.

Another important aspect that was not explored due to time constraints is the impact of differential privacy on model performance. Differential privacy is crucial for ensuring data privacy, a fundamental principle of federated learning. Investigating how the addition of random noise to the generated data points affects the performance of downstream tasks is essential. Such research would be valuable for understanding the trade-offs between privacy and model accuracy, facilitating the adoption of generative data augmentation in industry.

Lastly, our study primarily focused on federated learning frameworks where relatively fewer clients participate, commonly found in the medical domain. This approach overlooks scenarios with a large number of participating clients, such as mobile keyboard next-word recommenders [12] [26]. It is unlikely that our results will directly translate to large federated learning systems. Future work should explore the effects of data augmentation in environments with many clients to better understand its scalability and applicability across different federated learning contexts.

7 Responsible Research

Federated learning is being increasingly used for data-sensitive fields, such as the medical field, where a privacy breach would be disastrous. Therefore, it follows that the ethical implications of federated learning usage and more specifically privacy losses in generative modelling should be considered with relation to our research project. Furthermore, we must uphold the principles stated in the Dutch code of conduct for research integrity [16] by clearly documenting our data, analysis, and methodology.

7.1 Ethical Implications of Using Generative Augmentation for Federated Learning in Privacy Sensitive Scenarios

The use of generative augmentation in federated learning, particularly in privacy-sensitive scenarios, poses significant ethical implications. A primary concern is the risk of information leakage. Information leakage occurs when private data can be partially reconstructed through various methods. This risk is exacerbated when a client possesses only a few

data points, making their information more susceptible to re-identification.

Moreover, sharing variational autoencoder (VAE) models introduces additional vectors for potential data leaks. There is a risk that some generated data might exactly match the data on which the generative model was trained [10]. This scenario creates a tension between achieving high model performance and preserving data privacy. High-performing models may inadvertently memorize and regenerate specific training data, thereby compromising privacy.

One viable solution to mitigate these risks is the implementation of differential privacy. Differential privacy, as described by [8], provides a framework for quantifying and managing privacy loss. It works by introducing random noise to the shared information, which, in the context of our research, would be the weights of the generative models. By carefully calibrating the amount of noise added, stakeholders can balance the trade-off between model accuracy and the degree of privacy preservation. This approach allows for a more controlled and transparent parameterization of privacy, ensuring that sensitive data remains protected even as models are trained collaboratively.

7.2 Reproducibility and Repeatability

To ensure the reproducibility of our experiments in Section 4, we have published our code (which consists of plot generation, as well as our modifications to the federated learning simulation framework originally developed by [19]) on Github² with the fully open source MIT License, with which allows researchers to reproduce and build upon our results. The exact commands we used for each run and each experiment are also available in the README file on our GitHub repository. However, it is important to note, that while our code is available, future researchers may not achieve identical results due to the randomness present in the training of ML models.

A Appendix

I acknowledge the use of ChatGPT 3.5 and ChatGPT 4 (OpenAI, <https://chat.openai.com>) to rephrase initial drafts, generate basic skeleton code for plotting, and proofread my final draft.

Prompt: Reword this to flow smoother:

The number of local epochs and global communication rounds were set to 1 and 15 respectively since performance didn't improve much for any augmentation method by increasing them.

Response:

The number of local epochs and global communication rounds were set to 1 and 15, respectively, as increasing them did not significantly enhance the performance for any augmentation method.

Prompt: Make this flow smoother:

²<https://github.com/Lyadalachanchu/Effect-of-Different-Data-Augmentation-Strategies-on-Performance-In-Federated-Learning-Systems>

Synthetic imputation also shows steady performance improvement even with the more balanced client datasets as evidenced in Figure ref{minority-test-f1-imputation}; whereas with uniform imputation performance seems to stagnate.

Response:

Synthetic imputation demonstrates steady performance improvement, even with more balanced client datasets, as shown in Figure ref{minority-test-f1-imputation}. In contrast, uniform imputation results in performance stagnation.

Prompt: Make this more concise:

Our experiments were performed on the binarized MNIST dataset with the ExquisiteNet classifier being trained in a federated manner. The results indicate a notable difference in quality between the synthetic and original MNIST datasets, leading to poorer classification performance. This problem is evident in the finetuning experiments, where classifiers trained on synthetic data showed significant accuracy degradation across all values. Specifically, the pretrained classifiers, trained only with synthetic data, performed worse compared to other strategies. However, incorporating both original and synthetic data during training improved performance, especially for heavily imbalanced data.

Response:

Our experiments utilized the binarized MNIST dataset with the ExquisiteNet classifier trained in a federated manner. The results reveal a significant quality gap between synthetic and original MNIST datasets, resulting in diminished classification performance. Fine-tuning experiments demonstrate substantial accuracy degradation across all values for classifiers trained solely on synthetic data. However, integrating both original and synthetic data during training enhances performance, particularly in heavily imbalanced datasets.

Prompt: Turn this list of bullet points into a well-written section:

Data heterogeneity is a significant challenge that impedes the effectiveness of practical federated learning systems, often leading to substantial performance degradation.

- Most important results (aka. your main contributions)
 - In this project, two data-based methods, synthetic uniform imputation and synthetic minority imputation, were investigated. A framework-based approach that involves locally fine-tuning a classifier trained centrally on synthetic data was also explored.

- Combination of original and synthetic data seem to outperform only training on only original data or only synthetic data for less balanced label distributions
- There seems to be an optimal amount of synthetic data imputation, after which performance is similar if not slightly better than with no imputation
- It was reinforced that using only synthetic data for classifier training leads to significantly worse performance for downstream tasks.

Response:

Data heterogeneity is a significant challenge that impedes the effectiveness of practical federated learning systems, often leading to substantial performance degradation. This project addresses this issue by investigating two data-based methods: synthetic uniform imputation and synthetic minority imputation. Additionally, a framework-based approach was explored, which involves locally fine-tuning a classifier that was initially trained centrally on synthetic data.

The key findings of the project are as follows:

1. Combining original and synthetic data outperforms training exclusively on either original data or synthetic data, especially in scenarios with less balanced label distributions.
2. There appears to be an optimal amount of synthetic data imputation, beyond which performance stabilizes and may slightly improve compared to no imputation.
3. It was reaffirmed that using solely synthetic data for classifier training leads to significantly worse performance in downstream tasks.

These results highlight the potential of synthetic data augmentation to mitigate the effects of data heterogeneity in federated learning, offering insights into optimal strategies for improving model performance.

Prompt:

How can I get the total number of times a label appeared:

```
generated_train_dataset = impute_cvae_minority(k=1000, trained_cvae=trained_cvae, initial_dataset=train_dataset)

'post_label_counts = [label for _, label in generated_train_dataset]'
'post_label_counts = Counter(post_label_counts)
```

Response:

To get the total number of times a label appeared in the generated dataset, you can use the 'Counter' from the 'collections' module. Heres

how you can modify your code to count the occurrences of each label in the concatenated dataset:

```
from collections import Counter

def impute_cvae_minority(k, trained_cvae,
    initial_dataset):
    # Get the label distribution from the initial
    dataset
    label_distribution = get_label_distribution(
        initial_dataset)

    # Invert the distribution to prioritize
    minority labels
    max_prob = max(label_distribution.values())
    inverted_distribution = {label: max_prob / prob
        for label, prob in label_distribution.items()}
    total_inverted_prob = sum(inverted_distribution
        .values())
    normalized_distribution = {label: prob /
        total_inverted_prob for label, prob in
        inverted_distribution.items()}

    # Prepare the transformation
    apply_transform = transforms.Compose([
        transforms.ToTensor(),
        transforms.Lambda(lambda x: torch.round(x))
    ])
    # Binarize the images

    # Generate dataset
    generated_dataset = []
    generated_labels = random.choices(list(
        normalized_distribution.keys()), weights=
        normalized_distribution.values(), k=k)

    for label in generated_labels:
        generated_image = trained_cvae.
            generate_data(n_samples=1, target_label=label).
            squeeze(1)
        multiplier = 1.0 / generated_image.max().
            item()
        transformed_image = torch.round(
            generated_image * multiplier)
        generated_dataset.append((transformed_image
            , label))

    # Create a new dataset by concatenating the
    initial and generated datasets
    concatenated_dataset = torch.utils.data.
        ConcatDataset([initial_dataset,
            generated_dataset])

    # Count the occurrences of each label
    post_label_counts = [label for _, label in
        concatenated_dataset]
    post_label_counts = Counter(post_label_counts)

    return concatenated_dataset, post_label_counts

# Example usage (assuming you have a trained CVAE
model and initial dataset)
generated_train_dataset, post_label_counts =
    impute_cvae_minority(k=1000, trained_cvae=
```

```
trained_cvae, initial_dataset=train_dataset)
print(post_label_counts)
```

This code snippet defines a function to impute data using a CVAE while prioritizing minority labels, then counts and prints the occurrences of each label in the resulting concatenated dataset.

References

- [1] Josephine Akosa. Predictive accuracy: A misleading performance measure for highly imbalanced data. In *Proceedings of the SAS Global Forum 2017 Conference*. SAS Institute Inc., 2017.
- [2] Monik Raj Behera, Sudhir Upadhyay, Suresh Shetty, Sudha Priyadarshini, Palka Patel, and Ker Farn Lee. Fedsyn: Synthetic data generation using federated learning. 2022.
- [3] Subrato Bharati, M. Rubaiyat Hossain Mondal, Prajoy Podder, and V.B. Surya Prasath. Federated learning: Applications, challenges and future directions. *International Journal of Hybrid Intelligent Systems*, 18(1-2):19–35, May 2022.
- [4] Huancheng Chen and Haris Vikalo. Federated learning in non-iid settings aided by differentially private synthetic data, 2023.
- [5] Diletta Chiaro, Edoardo Prezioso, Michele Ianni, and Fabio Giampaolo. FI-enhance: A federated learning framework for balancing non-iid data with augmented and shared compressed samples. *Information Fusion*, 98:101836, 2023.
- [6] Onat Dalmaz, Usama Mirza, Gökberk Elmas, Muzaffer Özbey, Salman UH Dar, Emir Ceyani, Salman Avestimehr, and Tolga Çukur. One model to unite them all: Personalized federated learning of multi-contrast mri synthesis. 2022.
- [7] Li Deng. The MNIST database of handwritten digit images for machine learning research [best of the web]. *IEEE Signal Process. Mag.*, 29(6):141–142, November 2012.
- [8] Cynthia Dwork and Aaron Roth. The algorithmic foundations of differential privacy. *Found. Trends Theor. Comput. Sci.*, 9(3-4):211–407, 2013.
- [9] Ebba Ekblom, Edvin Listo Zec, and Olof Mogren. Effgan: Ensembles of fine-tuned federated gans, 2022.
- [10] Ashkan Vedadi Gargary and Emiliano De Cristofaro. A systematic review of federated generative models, 2024.
- [11] Tianyu Han, Sven Nebelung, Christoph Haarburger, Nicolas Horst, Sebastian Reinartz, Dorit Merhof, Fabian Kiessling, Volkmar Schulz, and Daniel Truhn. Breaking medical data sharing boundaries by using synthesized radiographs. *Science Advances*, 6(49), December 2020.

- [12] Andrew Hard, Chloé M Kiddon, Daniel Ramage, Françoise Beaufays, Hubert Eichner, Kanishka Rao, Rajiv Mathews, and Sean Augenstein. Federated learning for mobile keyboard prediction, 2018.
- [13] Eunjeong Jeong, Seungeun Oh, Jihong Park, Hyesung Kim, Mehdi Bennis, and Seong-Lyun Kim. Hiding in the crowd: Federated data augmentation for on-device learning. *IEEE Intelligent Systems*, 36(5):80–87, 2020.
- [14] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. December 2014.
- [15] Diederik P Kingma and Max Welling. Auto-encoding variational bayes. 2022.
- [16] KNAW, NFW, NWO, TO2-federatie, Vereniging Hogescholen, and VSNU. Netherlands code of conduct for research integrity, 2018. Available at https://www.vsnul.nl/en_GB/netherlands-code-of-conduct-for-research-integrity.
- [17] Claire Little, Mark Elliot, and Richard Allmendinger. Federated learning for generating synthetic data: a scoping review. *International Journal of Population Data Science*, 8(1), October 2023.
- [18] Xiaodong Ma, Jia Zhu, Zhihao Lin, Shanxuan Chen, and Yangjie Qin. A state-of-the-art survey on solving non-IID data in federated learning. *Future Gener. Comput. Syst.*, 135:244–258, October 2022.
- [19] H Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Agüera y Arcas. Communication-efficient learning of deep networks from decentralized data. February 2016.
- [20] Suman Ravuri and Oriol Vinyals. Classification accuracy score for conditional generative models. May 2019.
- [21] Ruslan Salakhutdinov and Iain Murray. On the quantitative analysis of deep belief networks. In *Proceedings of the 25th international conference on Machine learning - ICML '08*, ICML '08. ACM Press, 2008.
- [22] Felix Sattler, Klaus-Robert Müller, and Wojciech Samek. Clustered federated learning: Model-agnostic distributed multi-task optimization under privacy constraints. *CoRR*, abs/1910.01991, 2019.
- [23] Myungjae Shin, Chihoon Hwang, Joongheon Kim, Jihong Park, Mehdi Bennis, and Seong-Lyun Kim. XOR mixup: Privacy-preserving data augmentation for one-shot federated learning. *CoRR*, abs/2006.05148, 2020.
- [24] Taylor G. Smith and Jason M. White. Tg-smith61591/smrt: Handle class imbalance intelligently by using variational auto-encoders to generate synthetic observations of your minority class., 2017.
- [25] Kihyuk Sohn, Honglak Lee, and Xinchen Yan. Learning structured output representation using deep conditional generative models. In C. Cortes, N. Lawrence, D. Lee, M. Sugiyama, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 28. Curran Associates, Inc., 2015.
- [26] Differential Privacy Team. Learning with privacy at scale, Dec 2017.
- [27] Paul Voigt and Axel von dem Bussche. *The EU general data protection regulation (GDPR)*. Springer International Publishing, Cham, Switzerland, 1 edition, August 2017.
- [28] Mikhail Yurochkin, Mayank Agarwal, Soumya Ghosh, Kristjan Greenewald, Trong Nghia Hoang, and Yasaman Khazaeni. Bayesian nonparametric federated learning of neural networks. May 2019.
- [29] Yue Zhao, Meng Li, Liangzhen Lai, Naveen Suda, Damon Civin, and Vikas Chandra. Federated learning with non-IID data. June 2018.
- [30] Hangyu Zhu, Jinjin Xu, Shiqing Liu, and Yaochu Jin. Federated learning on non-IID data: A survey. June 2021.