Addressing Statistical Heterogeneity through Generative Similarity-Based Comparison in Federated Learning

Aggregation Weight Modifications Using Latent Space Insights

Henry Page[1]

Supervisors: Dr. David Tax[1]**, Swier Garst**[1]

[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: Henry Page
Final project course: CSE3000 Research Project
Thesis committee: Dr. David Tax, Swier Garst, Dr. Alexios Voulimeneas

Abstract

Federated Learning (FL), is a distributed learning approach where multiple clients collaboratively train a model whilst maintaining data security and privacy. One significant challenge in FL that must be addressed is statistical heterogeneity within the data. This occurs because data across different clients may not come from the same distribution, potentially leading to sub-optimal performance. To address this, we examine how insights gained from a generative model's latent space can mitigate these problems by adjusting the aggregation weight (influence) assigned to each client during the training process. We leverage information derived from a Variational Autoencoder (VAE) trained in a federated manner and propose a method to modify the aggregation weight of each client in FL. This method considers local discrepancies, resulting from differences between the local latent space distributions and global latent space distributions, together with the dataset sizes of each client. Experiments were conducted on the MNIST and Fashion-MNIST datasets. Our results indicate that our method enhances the model's performance by up to 6.76% in the best case, in terms of reducing the average test VAE loss and accelerating the convergence of the VAE in scenarios characterised by severe data imbalances among clients. It worsens performance when all clients have an equal level of imbalance. The source code for our research is available at https://github.com/FederatedRP2024Delft/Federated-Learning-PyTorch-Weight-Modification.

1 Introduction

Federated Learning (FL), initially proposed by McMahan et al. [1] is a Machine Learning (ML) paradigm which sees a set of massively distributed clients collectively train a unified global model. In short, each client trains a model on their own dataset, aiming to reduce the training error specific to their dataset. Afterwards, the model parameters from all (selected) clients are aggregated to make a new model. The enhanced model is shared with all clients who then undergo additional rounds of training to further refine the global model by reiterating the previously outlined procedure (Figure 1). This process is repeated across multiple epochs, referred to as communication rounds in a FL context.

This approach eliminates the need for clients to share their data, allowing for the development of ML models whilst ensuring data security and privacy. Additionally, the vast size of datasets residing on these clients often makes it impractical to transmit this data to a central location for training purposes due to resource constraints [1], [2]. These attributes render FL methods especially appealing for applications in sensitive domains such as healthcare [2].

Training a model in a federated manner introduces certain challenges, one of which is the statistical heterogeneity that

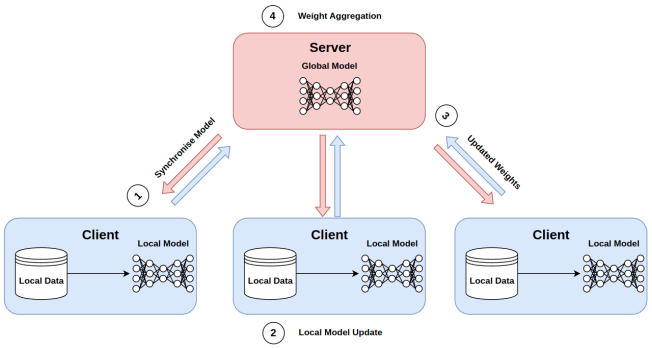


Figure 1: Illustration of FL that shows weight distribution among clients, local training, and subsequent weight aggregation.

arises from using FL. Typically in FL, the data distribution among clients varies, which contradicts the conventional assumption that data are independent and identically distributed (IID) [3]. In this thesis, we focus on a type of statistical heterogeneity, referred to as category distribution heterogeneity. This occurs when some clients possess a significant number of samples from specific classes, whilst others do not have a comparable amount of samples from those same classes. This is problematic as "different local models are optimised towards different local objectives, causing divergent optimisation directions" [4, p. 1]. This discrepancy can lead to performance issues such as a decrease in accuracy and slower model convergence [3].

Optimisation approaches like the widely accepted *FedAvg* algorithm have been proposed to facilitate the FL process [1]. This algorithm, however, has been observed to diverge in scenarios where the data across clients is not IID [1], [3]. Various studies have investigated different strategies to address this issue. Examples include: utilising clustering algorithms to categorise similar clients and incorporating regularisation terms to constrain local models [5]. Our research focuses on one specific approach: adjusting the aggregation weights of the clients, a method that has demonstrated enhanced performance [4], [5].

This thesis explores the less commonly researched subject of adjusting aggregation weights in FL by investigating generative models' latent spaces. We examine the properties of a generative model known as a Variational Autoencoder (VAE), which learns a compressed latent representation of the original data. This investigation is motivated by the ability to use information from the latent space of each client to infer the similarity between the client's data distribution and the overall global data distribution, thereby allowing us to weigh the importance of each client accordingly. This raises the question: *how does adapting aggregation weights in FL, based on the differences between the global latent space and locally encoded sample distributions of a VAE, affect model performance and convergence*? This thesis provides insights into the feasibility of training a VAE in a federated manner, such that the difference between the locally encoded sample distribution of each client and the global latent space distribution can be used to re-modify the aggregation weights of clients

when retraining said VAE from scratch.

This thesis begins by providing additional context pertinent to the problem in Section 2. Section 3 discusses the proposed method for adjusting client weights. Following this, Section 4 details the experimental setup and the results obtained. Section 5 analyses these results, in the context of existing literature. This thesis concludes with Section 6 and Section 7, where recommendations for future research are given alongside a discussion regarding the ethical considerations inherent to this subject matter.

2 Background

This section offers important contextual knowledge which is relevant for Section 3. Subsection 2.1 discusses more background knowledge about *FedAvg* and an improved version of *FedAvg* known as *FedDisco*. Subsection 2.2 discusses some background information related to the generative model which was used in our experiments.

2.1 FedAvg and FedDisco

FL aims to minimise the finite-sum objective function seen in Equation 1:

$$\min_{\omega} \sum_{k=1}^{K} p_k F_k(\omega) \tag{1}$$

where K represents the number of clients, F_k denotes the local objective function for client k, and p_k indicates the aggregation weight of client k. The algorithm starts by initialising global weights ω_0. Client k is trained in round t with the global weights of ω_{t-1} for E local epochs. To determine ω_{t+1}, the weighted average is taken over the updated model parameters of each client ω_{t+1}^k weighed by p_k. In *FedAvg*, this is equivalent to the relative dataset size of client k [1]. It is important to note, that the original *FedAvg* algorithm uses a subset of clients for local updates. This thesis uses all K clients across all communication rounds in experiments for simplicity, as reflected in Algorithm 1.

The best performance in FL is realised when a larger weight is assigned to better-performing local models. Empirical observations indicate that the discrepancy level of each client's local dataset offers a more precise reflection of local model performance than the relative dataset size [4]. By minimising the convergence error bound for *FedAvg*, an optimised aggregation weight can be derived (Equation 2). It considers a scalar discrepancy value d_k that corresponds to how far a client's local data distribution is from the (ideal) global data distribution. This discrepancy is considered along with the relative dataset size n_k, and hyperparameters α and b, which regulate the influence of d_k in determining the weight p_k [4].

$$p_k \propto n_k - \alpha \cdot d_k + b \tag{2}$$

Following this observation, *FedDisco* was proposed by Ye et al. [4] as an improvement to *FedAvg*, with the key difference being in the determination of p_k (Equation 3):

Algorithm 1 *FedAvg*. K denotes the number of clients, T is the number of communication rounds, E is the number of local epochs, B is the local batch size, η is the learning rate.

function FEDAVG(T, E, B, K, η)
  initialise ω_0
  for each round $t = 1, 2, \dots T$ **do**
    for each client $k = 1, 2 \dots K$ **do**
      $\omega_{t+1}^k \leftarrow \text{ClientUpdate}(k, \omega_t, E, B, \eta)$
    end for
    $\omega_{t+1} \leftarrow \sum_{k=1}^{K} p_k \cdot \omega_{t+1}^k$
  end for
  return ω_{T+1}
end function
function CLIENTUPDATE(k, ω, E, B, η)
  $\mathcal{B} \leftarrow$ (split $\mathcal{P}_k$ into batches of size B)
  for each local epoch i from 1 to E **do**
    for batch $b \in \mathcal{B}$ **do**
      $\omega \leftarrow \omega - \eta \nabla \ell(\omega; b)$
    end for
  end for
  return ω
end function

$$p_k = \frac{(n_k - \alpha \cdot d_k + b)_+}{\sum_{m=1}^{K} (n_m - \alpha \cdot d_m + b)_+} \tag{3}$$

where ReLU (denoted by $(x)_+$) is used to eliminate negative weights [6]. *FedDisco* calculates the local discrepancy d_k by applying a statistical metric function (such as KL divergence) "between its local category distribution and the hypothetically aggregated global category distribution", which they assume to be a uniform distribution [4, p. 5]. The rationale is to impose greater penalties on clients with imbalanced datasets compared to those with balanced datasets, where the local data has similar frequencies across all classes. This is why the statistical distance to a uniform distribution is used [4]. In Section 3, we use Equation 3 alongside our customised approach of determining local discrepancy which eliminates the need to assume the shape of the global data distribution. Instead, it draws from specific insights gained from training a VAE in a federated fashion.

2.2 β-Variational Autoencoder

A Variational Autoencoder (VAE), is a type of generative model represented as a neural network that encodes input data into a compressed latent space and then decodes it back to the original data. Unlike normal autoencoders, which are deterministic models, VAEs represent the latent space with a continuous probability distribution [7]. This key difference allows VAEs to generate new data points by sampling from said distribution, a capability which is not inherently possible with regular autoencoders.

The training process of a VAE involves minimising a loss function that combines the reconstruction loss (as defined for regular autoencoders) with the KL divergence between the

encoded sample(s) and a predefined prior distribution $p(z)$. This distribution is usually defined as a standard normal distribution, $\mathcal{N}(0, I)$ [7]. β-VAE is a type of VAE which extends this idea but puts a coefficient β in front of the KL-loss term which assigns a higher weight to the KL-Loss, as shown in Equation 4 [7], [8]. The parameters of the decoder and encoder networks are represented by θ and ϕ respectively. The reconstruction loss is the expected log-likelihood of obtaining a data sample $x^{(i)}$ being reconstructed from a latent vector $z^{(i)}$, which is sampled from the approximate posterior distribution $q_\phi(z|x^{(i)})$. Here, J denotes the dimensionality of the latent variable, and $\odot$ denotes element-wise multiplication. The encoded values for sample $x^{(i)}$ are denoted by $\mu_j^{(i)}$ and $\sigma_j^{(i)}$. They represent the outputs of the encoder's last hidden layer in an arbitrary latent dimension j.

$$\mathcal{L}(\theta, \phi; x^{(i)}) \simeq \underbrace{\mathbb{E}[\log p_\theta(x^{(i)}|z^{(i)} \sim q_\phi(z|x^{(i)}))]}_{\text{Reconstruction Loss}} + \underbrace{\frac{\beta}{2}\sum_{j=1}^{J}(1 + \log((\sigma_j^{(i)})^2) - (u_j^{(i)})^2 - (\sigma_j^{(i)})^2)}_{\text{KL-Loss}}$$
$$\text{where } z^{(i)} = \mu^{(i)} + \sigma^{(i)} \odot \epsilon \text{ and } \epsilon \sim \mathcal{N}(0, I) \quad (4)$$

It is crucial to note that the loss function penalises the model during training if an encoded sample does not closely represent the predefined prior distribution due to the KL-Loss regularisation term as seen in Equation 4. Furthermore, when $\beta > 1$, "the model is pushed to learn a more efficient latent representation of the data, which is disentangled if the data contains at least some underlying factors of variation that are independent" [8, p. 3]. Based on this information, it follows that under an IID setting, the learned prior distribution $q(z)$ closely approximates $\mathcal{N}(0, I)$. In Section 3, we discuss how this property can be used in a federated setting in tandem with what was discussed in Subsection 2.1.

3 Determining Client Weights

In this section, we outline the contribution of this thesis and introduce a method that leverages the concepts from Section 2 to adjust client weights.

From the observations in Section 2, we understand that since the global latent space closely resembles the predefined prior distribution, a client's dataset that closely represents the global data distribution will produce locally encoded samples that align well with the prior distribution. Conversely, if a client has a highly imbalanced dataset, it will not match the predefined prior distribution as well.

We conducted a preliminary trial to assess the impact of an imbalanced dataset on a client's sample encoding distribution. For this trial, we trained a standard VAE in a federated manner on the MNIST dataset [9]. The setup included four clients: two with IID local data distributions and two with Non-IID local data distributions created using $Dir(0.5)$ (Figure 2). Empirical observations highlight two key points

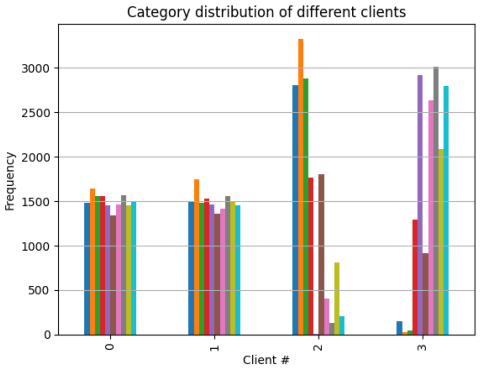
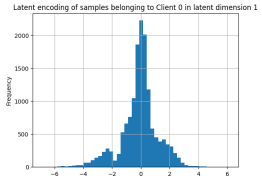
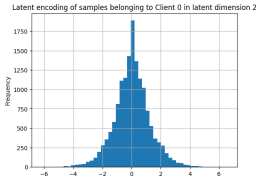


Figure 2: Two clients have IID data distributions, while the other two follow a $Dir(0.5)$ distribution. Each coloured bar represents one of the ten classes in MNIST.

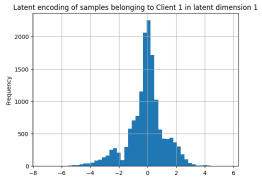
about the two clients with IID distributions. Across both latent dimensions, their distributions closely match the prior distribution, except for a small anomalous gap on the left tail in dimension 2 (Figure 3). The distributions of these two clients are also virtually identical, which is to be expected. The other two clients with imbalanced distributions depict different characteristics. In dimension 1, their distributions do not resemble a normal distribution. In dimension 2, whilst the shape of the distribution is somewhat similar to a normal distribution, it is not centred around zero (Figure 4).



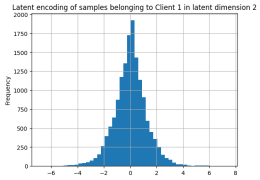
(a) Client 0, Dimension 1



(b) Client 0, Dimension 2



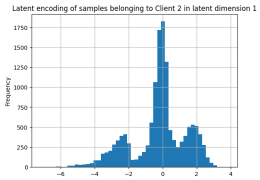
(c) Client 1, Dimension 1



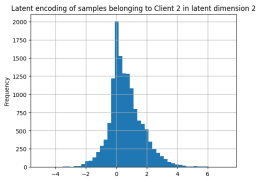
(d) Client 1, Dimension 2

Figure 3: Encoding sample distributions of clients with IID data distributions

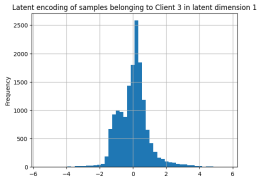
We hypothesise that we can leverage this small yet significant insight to adjust client weights based on the resemblance between each client's locally encoded sample distribution and the prior distribution $\mathcal{N}(0, I)$. After training a β-VAE in a federated manner, to determine the new weight p_k for client k, we will follow these steps: First, we will encode all local samples on client k by performing a forward pass through the trained encoder network. Next, we will evaluate each latent dimension individually (as each latent dimension is independent [8]) and compute the Wasserstein distance [10] between



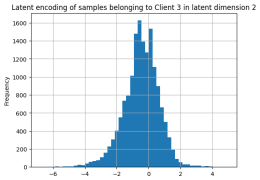
(a) Client 2, Dimension 1



(b) Client 2, Dimension 2



(c) Client 3, Dimension 1



(d) Client 3, Dimension 2

Figure 4: Encoding sample distributions of clients with $Dir(0.5)$ data distributions

a $\mathcal{N}(0, 1)$ distribution and the values of the locally encoded samples for that specific latent dimension. To reiterate, we focus on the statistical distance to $\mathcal{N}(0, 1)$, which represents the prior distribution $p(z)$. The rationale is to measure the extent to which the locally encoded sample distribution deviates from $p(z)$. The average of these distances across all latent dimensions will be calculated to determine the local discrepancy factor d_k. As discussed, we use d_k in Equation 3 to determine the new weights of each client. This process is outlined in Algorithm 2.

We opted to use Wasserstein distance over KL divergence for 3 reasons. Wasserstein distance is a true distance metric, unlike KL divergence, which fails to meet the symmetry property. This is significant because it ensures Wasserstein distance forms a metric space [11]. Furthermore, the Wasserstein distance between two probability distributions represents the minimum amount of "*work required to transport the probability mass from one distribution state to another. This characteristic provides a smooth and significant representation of the distance between distributions, making it a good candidate in domains where analysing the similarity of outcomes is more relevant than an exact likelihood matching*" [11, p. 3]. This is crucial for our use case, as we prioritise the notion of similarity between the distribution of local sample encodings and the global prior over the differences in likelihood at specific points within said distributions. Although not an inherent advantage of Wasserstein over KL divergence, it is worth noting that Wasserstein distance only takes values in the range $[0, \infty)$. This is important as when Equation 3 was proposed, the authors determined d_k using either the L2 difference or the KL divergence [4], both of which have the range $[0, \infty)$. Therefore, using the Wasserstein distance in this context is justified.

In Section 4, we implement this method and outline the experiment conducted in this thesis to evaluate its practicality and effectiveness in achieving performance improvements over how clients are conventionally weighed in the *FedAvg* algorithm.

Algorithm 2 Proposed algorithm used to determine new weights. K denotes the number of clients, Enc_ϕ is used to denote the trained encoder network with learned parameters ϕ. α and b are hyperparameters.

function DETERMINE_NEW_WEIGHTS(K,Enc_ϕ, α, b)
  for each client $k = 1, 2 \dots K$ **do**
    $\mathcal{E}^{m \times n} \leftarrow$ Generate embeddings of k's local samples through forward pass on Enc_ϕ
    $d_k \leftarrow 0$
    for each dim $\in \mathcal{E}^\intercal$ **do** ▷ Each latent dimension
      $\mathcal{D} \leftarrow$ Wasserstein_Distance(dim, $\mathcal{N}(0, 1)$)
      $d_k \leftarrow d_k + \frac{\mathcal{D}}{n}$
    end for
    $w_k \leftarrow (n_k - \alpha \cdot d_k + b)_+$ ▷ Equation 3
  end for
  $d_K \leftarrow \sum_{k=1}^{K} w_k$
  for $k = 1, 2, \dots K$ **do**
    $w_k \leftarrow \frac{w_k}{d_K}$
    $p_k \leftarrow w_k$
  end for
end function

4 Experiments

Experimental setups are shown in Subsection 4.1 and results are presented in section Subsection 4.2.

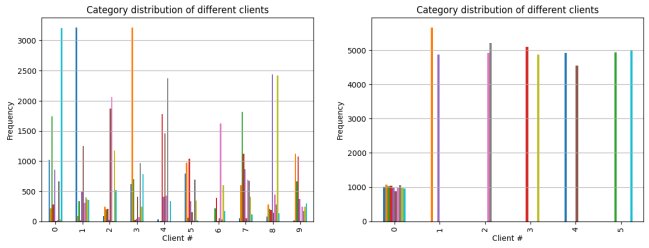
4.1 Experimental Setup

Datasets: We consider two image datasets. Namely, MNIST [9], which is a dataset of handwritten digits and Fashion-MNIST which is a dataset of fashion item images [12]. Both datasets contain $70,000$ 28×28 grey-scale images with $60,000$ training and $10,000$ test samples.

Data Preprocessing: For all experiments on both datasets, we normalised the entire dataset to ensure that the feature values have a zero mean and unit variance. Specifically, for MNIST, we used a mean of 0.1307 and a standard deviation of 0.3081. For FMNIST, we used a mean of 0.2860 and a standard deviation of 0.3530.

Federated Scenarios: We will explore federated scenarios (data distributions) identical to those introduced by Ye et al. in the FedDisco paper [4]. We will focus on two scenarios denoted as NIID-1 and NIID-2. NIID-1 comprises 10 clients with a heterogeneous data distribution. This distribution is achieved by sampling $\delta_i \sim Dir_K(0.5)$ and assigning $\delta_{i,k}$ proportion of class i instances to client k [13]. NIID-2 consists of 5 biased clients, each containing data from 2 out of 10 classes, alongside 1 unbiased client which has data from all classes [3], [4]. The data distributions for these given scenarios can be seen in Figure 5.

Implementation Details: All experiments were conducted with a fixed setup consisting of 25 communication rounds, 10 local epochs, and a local batch size of 64 samples. Throughout these experiments, we adjusted the hyperparameter α, which corresponds to the weight attributed to the discrepancy factor when computing the new weight of p_k (Equation 3). Different values of α in the set $\{0.1, 0.5, 0.9\}$ were used in



(a) NIID-1 (b) NIID-2

Figure 5: Visualisations of the class distributions in NIID-1 and NIID-2

every trial. The hyperparameter b was kept constant at $b = 0$.

Model: The model used was a standard β-VAE ($\beta = 10.0$), trained with the Adam optimiser at a learning rate of $\eta = 0.001$. A discussion of the model's architecture, along with a schematic diagram, is provided in Appendix A. For the reconstruction loss (Equation 4), we used mean-squared error (MSE). It is important to clarify that instead of the usual MSE, where the average of all the squared differences is taken, we used the sum of the squared differences. This approach is standard for images, as the expectation of the log-likelihood is calculated as the sum of all squared differences for each pixel, rather than the average of the differences for each pixel.

Software and Hardware: We employed an open-source federated 'framework' built in PyTorch. We also adapted the framework by integrating the client weight modification algorithms discussed in this thesis. Both the original framework[1] and our modified version[2] are available to the public. The experiments were conducted on a computer running Ubuntu 22.04.04, equipped with an AMD Ryzen 5800H CPU and an Nvidia RTX 3050M GPU.

Metrics: The average VAE loss (Equation 4) will be evaluated on the global model using all samples from the test set at the end of each communication round. This will enable us to assess the impact of our method on the performance of the β-VAE and to examine whether our method improves convergence speed.

4.2 Results

We collected results for the experimental setups detailed in Subsection 4.1 and conducted five trials for each federated setting to minimise the chance of undetected anomalies. It is worth mentioning that for trials with $\alpha = 0.9$, some were not tested five times due to big local discrepancies at all clients, which led to each client being assigned an undefined weight (0 in the denominator of Equation 3). Nevertheless, the overall trend observed in the remaining trials was consistent. To contextualise these results, we compared them to a baseline β-VAE which corresponds to a β-VAE trained using the regular *FedAvg* algorithm (Algorithm 1) where the client's weight is determined by its relative dataset size ($p_k = n_k$).

[1]https://github.com/AshwinRJ/Federated-Learning-PyTorch

[2]https://github.com/FederatedRP2024Delft/Federated-Learning-PyTorch-Weight-Modification

To summarise the results for NIID-1, it is evident that the introduction of new weights leads to a decline in the model's effectiveness on both the MNIST and FMNIST datasets when training is completed. Nevertheless, a notable observation is that the modified weights yield better performance compared to the baseline during the initial communication rounds as α increases.

In the MNIST dataset (Figure 6), we see that for $\alpha = 0.1$, it performs similarly to the baseline and both of them improve steadily over communication rounds, with $\alpha = 0.1$ being slightly worse in terms of performance. For $\alpha = 0.5$, the model initially performs better than the baseline in the first four communication rounds but then converges prematurely. Even after five trials, the performance for $\alpha = 0.5$ varies, as indicated by the high standard deviation of the loss across all communication rounds. We were able to test $\alpha = 0.9$ once for this federated setting, however, it shows sub-optimal performance after 5 communication rounds converging to an average test loss around 539.68, whereas the baseline converges towards 517.35. In NIID-1, $\alpha = 0.1$ shows a high standard deviation across all communication rounds, indicating that its performance varies depending on the specific data distribution, suggesting that it potentially outperformed the baseline in certain trials.

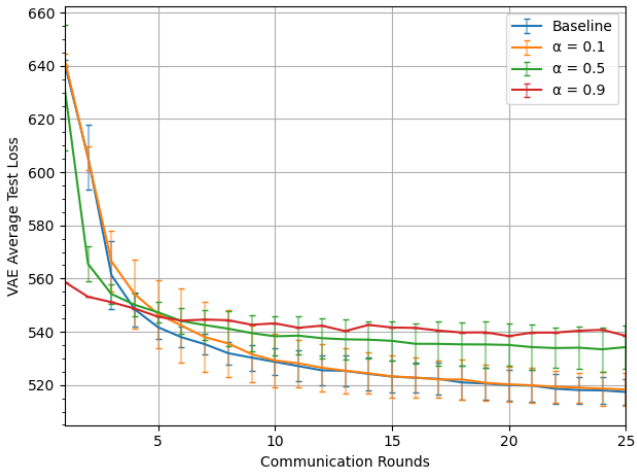


Figure 6: Average test loss difference over 25 communication rounds between a β-VAE with $\beta = 10.0$ and the same β-VAE retrained from scratch with modified weights across different α values under NIID-1 on the MNIST dataset

For the FMNIST dataset (Figure 7), similar behaviour was observed. Higher values of α helped the model converge relatively faster within the first three communication rounds. The improvement in the second communication round is evident, as the baseline shows an average VAE loss of 526.24, while the best-performing model ($\alpha = 0.9$) exhibits an average VAE loss of 479.71. When α values are too high, the model tends to converge towards a client's local objective, which is undesirable in the scenario where all clients are equally unbalanced (NIID-1).

For NIID-2, contrasting outcomes were observed. The pri-

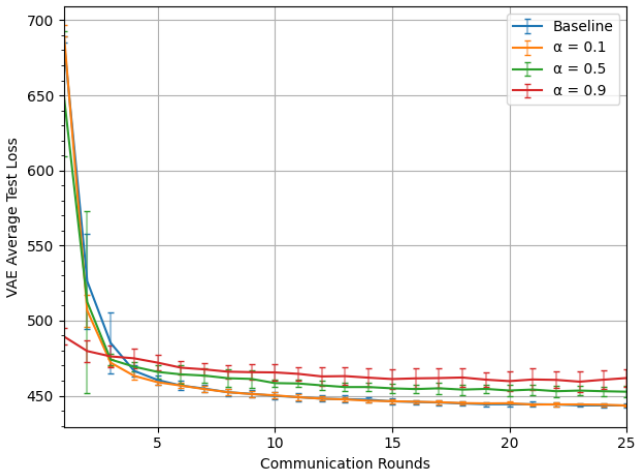


Figure 7: Average test loss difference over 25 communication rounds between a β-VAE with $\beta = 10.0$ and the same β-VAE retrained from scratch with modified weights across different α values under NIID-1 on the FMNIST dataset

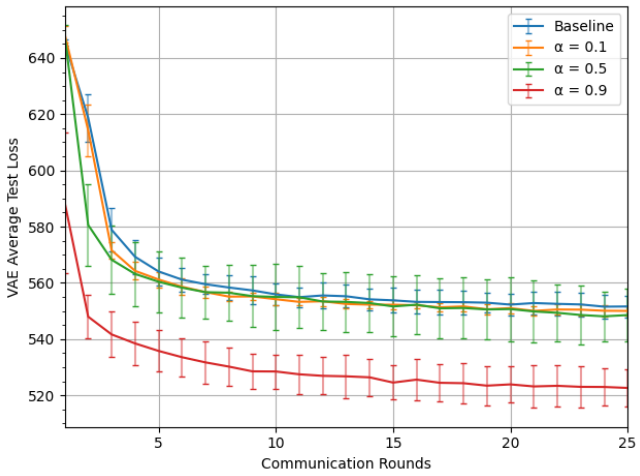


Figure 8: Average test loss difference over 25 communication rounds between a β-VAE with $\beta = 10.0$ and the same β-VAE retrained from scratch with modified weights across different α values under NIID-2 on the MNIST dataset

mary observation was that increasing α enhanced overall performance. The presence of a greater class imbalance within client datasets led to the baseline method demonstrating subpar performance. The baseline also exhibits the greatest variance across communication rounds. This is to be expected as each biased client only has two classes.

In the case of MNIST (Figure 8), there is not much of a difference between the baseline and the two α values $\alpha = 0.1$, $\alpha = 0.5$. Both settings performed better than the baseline, however, they all converged to around the same average test loss, which in the case of the baseline was 551.49, whilst the other federated settings achieved 550.01 and 548.53 respectively. In the case of $\alpha = 0.9$, the model performance was improved as it converged to an average test loss of 522.61. One notable similarity with NIID-1 is that $\alpha = 0.5$ exhibits a high error, indicating that in certain specific scenarios of data distribution, it could result in either better or worse performance.

The results aligned with our hypotheses when we tested the same federated scenario using the FMNIST dataset (Figure 9). One key difference from the MNIST dataset is that even lower values of $\alpha = 0.5$ yield a greater performance improvement than when $\alpha = 0.5$ was used on MNIST. This is evident in the second communication round, where the baseline shows an average test loss of 603.98, while $\alpha = 0.5$ and $\alpha = 0.9$ yield losses of 467.84 and 464.90, respectively. This performance is already better for the model with modified weights after just 2 communication rounds compared to the baseline model outcome after 25 rounds. Higher values of α promoted faster convergence than observed with the MNIST dataset. This is noticeable with $\alpha = 0.5$, where the average test loss difference between the first and second communication rounds is 64.1.

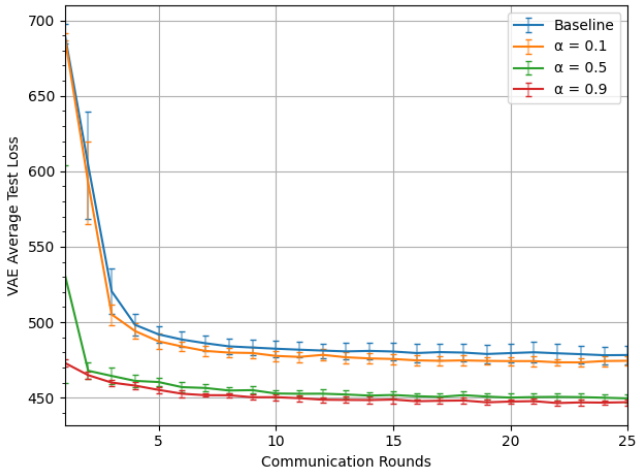


Figure 9: Average test loss difference over 25 communication rounds between a β-VAE with $\beta = 10.0$ and the same β-VAE retrained from scratch with modified weights across different α values under NIID-2 on the FMNIST dataset

5 Discussion

The results of our experiments can be contextualized within the broader landscape of FL and β-VAE optimisation. Unfortunately, a limited amount of research leverages latent space insights to adjust client weights in a federated setting, making it difficult to compare our results directly with other studies.

What we found aligns with existing literature, which suggests that FL models struggle with heterogeneous data distributions due to the non-IID nature of the data at each client [1], [3]. The high standard deviation observed in these settings underscores the challenge of achieving consistent performance across all communication rounds when client datasets are imbalanced.

Our results showed that the potential improvement in performance from using our method largely hinges on the choice of hyperparameters. Since we used a β-VAE with $\beta = 10.0$, the model was encouraged to learn a disentangled latent representation of the data, implying that latent factors are considered independent [14] and closely follow a $\mathcal{N}(0, 1)$ distribution regardless of how other latent variables are learned. This accounts for the observed enhancement in performance for federated settings when α was increased, as it resulted from penalising clients whose learned latent distribution did not closely match the predefined prior. Our findings also suggest that adjusting the α hyperparameter resulted in increased model performance, suggesting that fine-tuning this parameter is essential for optimal model performance. This claim is substantiated by comparing the results of NIID-1 and NIID-2, as comparing the same α values for the two different settings shows that it may not be optimal for that specific federated setting.

In scenarios with NIID-2, our method showed that there was a performance increase for all values of α that were tested. This outcome was anticipated. Unlike NIID-1, there is a client in the cluster with an IID dataset, leading to a smaller local discrepancy value compared to other clients, which only had 2 out of the 10 classes present. This was not the case in NIID-1, where all client datasets were similarly imbalanced. This could also serve as an explanation as to why very high values of α ($\alpha = 0.9$) lead to premature convergence and ultimately sub-optimal model performance. During the experiments, we observed that under NIID-1 and $\alpha = 0.9$, some clients were assigned a weight of 0.0. While this may be beneficial in the case of NIID-2 (as 1 client is guaranteed to have all classes), it could fail to consider some clients which mainly consist of classes that are encoded in a way that does not match the prior, which means that the global model effectively starts ignoring samples from certain classes. As a result, the model starts optimising towards the local objectives of a subset of clients which is also not optimal.

One way to remediate this is by increasing the b hyperparameter (Equation 3). This would ensure that all clients have non-zero weight. This was not tested in our experiments as we kept this hyperparameter fixed. Moreover, NIID-1 involved 10 users instead of 6, resulting in a more 'fragmented' dataset. In the case of NIID-2, all 60,000 training samples were utilised, in contrast with NIID-1, where the dataset division among clients limited the availability of these samples due to how they were drawn [13]. The selection of data preprocessing techniques applied to a dataset might additionally impact performance variability across various scenarios, although this aspect was not explored.

Further explanations for these results could involve the specific nature of the datasets used. The MNIST and FMNIST datasets, while both consisting of 28×28 grey-scale images, represent different types of classification tasks (digits vs fashion items). The varying complexities of these datasets likely influenced the performance improvements observed with different α values. This can be seen when we compare Figure 6 and Figure 8 against Figure 7 and Figure 9 respectively, as the variance in performance across communication rounds is lower for FMNIST. In NIID-2, the model trained using $\alpha = 0.5$ yields a 0.56% performance increase for MNIST, whereas in FMNIST it yields a 6.19% performance improvement after training. In the best-case scenario, the model trained with $\alpha = 0.9$ achieved a performance increase of 5.40% for MNIST and 6.76% for FMNIST.

6 Conclusions and Future Work

FL enables decentralised model training, preserving data privacy and improving resource usage by eliminating the need to transfer data to a central location for training [1]. However, it faces a fundamental challenge: there is no assurance that the data distributions across clients are independent and identically distributed (IID) [3]. This introduces statistical heterogeneity which is undesirable in an FL context.

This thesis investigated how the learned global latent space representation within a federally trained β-Variational Autoencoder (β-VAE) could be used to assess local discrepancies based on the distance between each client's local data distribution and the global data distribution. The local discrepancy of a client was determined by computing the Wasserstein distance between the prior distribution and the sample encoding distribution of each client. Following this, we proposed a novel way to adjust client weights during the federated training of a β-VAE. By integrating our version of local discrepancy with the optimised way of determining the aggregation weight of a client (p_k) as proposed by Ye et al. [4], the weights for each client were modified (using Equation 3) based on how well their encoding distribution matched the prior distribution $p(z)$.

We investigated how the new client weights as determined by our proposed method affected the model performance and convergence of the β-VAE, when retrained from scratch in two federated scenarios: NIID-1 and NIID-2. Our experiments demonstrate that as the data imbalance increased, our method enhanced model performance compared to the baseline (*FedAvg*). We also determined that our method works especially well in scenarios where some clients possessed a 'globally representative' dataset, but performed less effectively when all clients exhibited equal levels of imbalance. This discrepancy arises because a globally representative dataset tends to align more closely with the prior distribution in the latent space. In NIID-1, we observed instances where a high value for the α hyperparameter resulted in the weight of certain clients being set to zero, which resulted in samples from a subset of classes being disregarded, leading to premature convergence.

In terms of performance, we showed that under specific federated scenarios, the proposed method may produce less favourable outcomes compared to the baseline. However, under other scenarios, a performance improvement is observed. With respect to model convergence, we were able to show that this was very dependent on the chosen hyperparameters and the federated scenario. In NIID-1, most of our experiments exhibited convergence within the first few communication rounds, converging prematurely in some cases. Under NIID-2, convergence was achieved within a comparable number of communication rounds to the baseline across both datasets. However, this was accompanied by a lower average

VAE test loss.

As we conclude this thesis, we would like to point out several areas for further exploration that can be pursued.

As mentioned in Section 5, the proposed method was exclusively tested on MNIST and FMNIST, which demonstrated dissimilar model behaviour across the two datasets. An interesting avenue for exploration could be evaluating this method on more complex datasets such as CIFAR-10 and CIFAR-100 [15]. The primary limitation of MNIST and FMNIST lies in their grey-scale nature. CIFAR datasets offer the advantage of RGB images with three colour channels, presenting a more advanced learning task. Exploring more sophisticated model architectures could also be beneficial since our experiments were limited to testing a model with a two-dimensional latent space.

Our investigation focused on measuring the VAE loss across communication rounds, averaged across all test samples. While this approach provides a foundation when comparing model performance, it is essential to reiterate the advantages of a VAE over other generative models. By sampling from the latent space distribution of a VAE, one can generate new synthetic data. Even if our method effectively minimises the VAE loss, the quality of the synthetic images that can potentially be created from the model remains uncertain. An avenue for further exploration involves utilising synthetic images for downstream learning tasks and comparing performance metrics such as classification accuracy score (CAS) between the baseline method and our proposed method [16].

It is important to outline a significant limitation of the proposed method: its dependence on a pre-trained β-VAE. An adaptation that we could make is to periodically reweigh each client using Algorithm 2 after a specific number of communication rounds. This approach introduces a more practical means of integrating our method, potentially enhancing its utility. By doing so, the globally trained model would be encouraged to leverage its latent space information during training.

7 Responsible Research

When conducting research, it is crucial to consider the potential ramifications of ethical and societal impacts that may not have been initially anticipated, as understanding the broader implications of one's work is essential. Researchers must rigorously assess the validity of their research to ensure its reliability and accuracy. The following subsections address these issues.

7.1 Ethical Considerations

The ethical considerations that predominantly challenge FL revolve around issues of bias and privacy. One of the critical concerns is the risk of biased machine learning outcomes. For instance, our research might be applied in a way where clients with a higher proportion of data containing a minority class are assigned less weight. This practice is highly problematic as it significantly amplifies bias, undermining the fairness and inclusivity of the model. When applying machine learning research to real-world scenarios, it is crucial to ensure that all groups, particularly minority and underrepresented classes, are fairly represented and treated. If the techniques discussed in this paper are used on datasets where representation issues may exist, they must be enhanced to mitigate any inherent biases arising from the data itself.

Privacy concerns are important to address in FL. Even though data is not directly exchanged across clients, the model updates are. These updates can potentially be analysed to infer the underlying data of individual clients, thereby compromising privacy. This vulnerability highlights a significant challenge within the FL research domain.

7.2 Integrity and Validity

Ensuring the integrity and validity of data is crucial when conducting research. Results must be presented exactly as they were collected, without any alterations unless they are accompanied by a valid and transparent explanation. This thesis is committed to upholding this standard by presenting all results without omission. Several trials were conducted under consistent conditions. This method ensures that the findings are reliable and accurate. This research upholds a high standard of scientific integrity and validity by adhering to these principles. This also ensures that the thesis complies with the standards for good research practices as defined by the Netherlands Code of Conduct for Research Integrity [17].

7.3 Reproducibility

Reproducibility is a fundamental aspect of all research, ensuring that other researchers can verify the accuracy and precision of the findings [18]. If an experiment cannot be reproduced, the researcher must justify this limitation. Suppose a researcher understands that repeating the experiment under the same conditions might yield different results. In that case, they should explain any and all potential discrepancies that may arise during a repeated trial of the experiment. In this thesis, all experimental setups are detailed to facilitate replication. All code used to obtain the results were provided in the footnotes, ensuring complete transparency and enabling other researchers to reproduce the experiments accurately. Furthermore, all dependencies and their associated versions were included in the *requirements.txt* file, which can be found in the repository.

7.4 Use of AI

Acknowledging the role of generative AI tools such as GPT-3.5, GPT-4, and GPT-4o in our writing process is crucial. These tools were employed solely to enhance the conciseness and readability of the thesis through rephrasing; they were not utilised to produce any results. All outputs from these models were critically evaluated. The prompt that was used was: "Rephrase the text to improve clarity and conciseness".

References

[1] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y. Arcas, "Communication-Efficient Learning of Deep Networks from Decentralized Data," in *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics*, ser. Proceedings of Machine Learning Research, A. Singh and J. Zhu, Eds.,

vol. 54. PMLR, 20–22 Apr 2017, pp. 1273–1282. [Online]. Available: https://proceedings.mlr.press/v54/mcmahan17a.html

[2] C. Palihawadana, N. Wiratunga, A. Wijekoon, and H. Kalutarage, "Fedsim: Similarity guided model aggregation for federated learning," *Neurocomputing*, vol. 483, p. 432–445, Apr. 2022. [Online]. Available: http://dx.doi.org/10.1016/j.neucom.2021.08.141

[3] T. Li, A. K. Sahu, M. Zaheer, M. Sanjabi, A. Talwalkar, and V. Smith, "Federated optimization in heterogeneous networks," in *Proceedings of Machine Learning and Systems*, I. Dhillon, D. Papailiopoulos, and V. Sze, Eds., vol. 2, 2020, pp. 429–450. [Online]. Available: https://proceedings.mlsys.org/paper_files/paper/2020/file/1f5fe83998a09396ebe6477d9475ba0c-Paper.pdf

[4] R. Ye, M. Xu, J. Wang, C. Xu, S. Chen, and Y. Wang, "FedDisco: Federated learning with discrepancy-aware collaboration," in *Proceedings of the 40th International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, A. Krause, E. Brunskill, K. Cho, B. Engelhardt, S. Sabato, and J. Scarlett, Eds., vol. 202. PMLR, 23–29 Jul 2023, pp. 39 879–39 902. [Online]. Available: https://proceedings.mlr.press/v202/ye23f.html

[5] P. Qi, D. Chiaro, A. Guzzo, M. Ianni, G. Fortino, and F. Piccialli, "Model aggregation techniques in federated learning: A comprehensive survey," *Future Generation Computer Systems*, vol. 150, p. 272–293, Jan. 2024. [Online]. Available: http://dx.doi.org/10.1016/j.future.2023.09.008

[6] K. Fukushima, "Visual feature extraction by a multilayered network of analog threshold elements," *IEEE Transactions on Systems Science and Cybernetics*, vol. 5, no. 4, p. 322–333, 1969. [Online]. Available: http://dx.doi.org/10.1109/TSSC.1969.300225

[7] D. P. Kingma and M. Welling, "Auto-encoding variational bayes," *CoRR*, vol. abs/1312.6114, 2013. [Online]. Available: https://api.semanticscholar.org/CorpusID:216078090

[8] 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. [Online]. Available: https://openreview.net/forum?id=Sy2fzU9gl

[9] L. Deng, "The mnist database of handwritten digit images for machine learning research," *IEEE Signal Processing Magazine*, vol. 29, no. 6, pp. 141–142, 2012.

[10] R. L. Dobrushin, "Prescribing a system of random variables by conditional distributions," *Theory of Probability amp; Its Applications*, vol. 15, no. 3, p. 458–486, Jan. 1970. [Online]. Available: http://dx.doi.org/10.1137/1115049

[11] K. Faber, R. Corizzo, B. Sniezynski, M. Baron, and N. Japkowicz, "WATCH: wasserstein change point detection for high-dimensional time series data," *CoRR*, vol. abs/2201.07125, 2022. [Online]. Available: https://arxiv.org/abs/2201.07125

[12] H. Xiao, K. Rasul, and R. Vollgraf, "Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms," *ArXiv*, vol. abs/1708.07747, 2017. [Online]. Available: https://api.semanticscholar.org/CorpusID:702279

[13] M. Yurochkin, M. Agarwal, S. Ghosh, K. Greenewald, N. Hoang, and Y. Khazaeni, "Bayesian nonparametric federated learning of neural networks," in *Proceedings of the 36th International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, K. Chaudhuri and R. Salakhutdinov, Eds., vol. 97. PMLR, 09–15 Jun 2019, pp. 7252–7261. [Online]. Available: https://proceedings.mlr.press/v97/yurochkin19a.html

[14] E. Mathieu, T. Rainforth, N. Siddharth, and Y. W. Teh, "Disentangling disentanglement in variational autoencoders," in *Proceedings of the 36th International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, K. Chaudhuri and R. Salakhutdinov, Eds., vol. 97. PMLR, 09–15 Jun 2019, pp. 4402–4412. [Online]. Available: https://proceedings.mlr.press/v97/mathieu19a.html

[15] A. Krizhevsky, V. Nair, and G. Hinton, "Cifar-10 (canadian institute for advanced research)." [Online]. Available: http://www.cs.toronto.edu/~kriz/cifar.html

[16] S. Ravuri and O. Vinyals, "Classification accuracy score for conditional generative models," in *Advances in Neural Information Processing Systems*, H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, Eds., vol. 32. Curran Associates, Inc., 2019. [Online]. Available: https://proceedings.neurips.cc/paper_files/paper/2019/file/fcf55a303b71b84d326fb1d06e332a26-Paper.pdf

[17] KNAW, NFU, NWO, TO2-Federatie, Vereniging Hogescholen, and VSNU, "Nederlandse gedragscode wetenschappelijke integriteit," 2018. [Online]. Available: https://easy.dans.knaw.nl/ui/datasets/id/easy-dataset:110600

[18] D. B. Resnik and A. E. Shamoo, "Reproducibility and research integrity," *Accountability in Research*, vol. 24, no. 2, p. 116–123, Nov. 2016. [Online]. Available: http://dx.doi.org/10.1080/08989621.2016.1257387

A β-VAE Architecture

This appendix details the architecture of the β-VAE utilised in all experiments. A comprehensive schematic of the β-VAE can be seen in Figure 10.

Encoder Architecture: The encoder comprised three linear layers with ReLU activation functions for the hidden layers. The first layer reshaped the input vector x from 784 to 512 dimensions, the second layer reduced it from 512 to 256 dimensions, and the final layer further reduced it to 2 dimensions, corresponding to the dimensionality of the latent space.

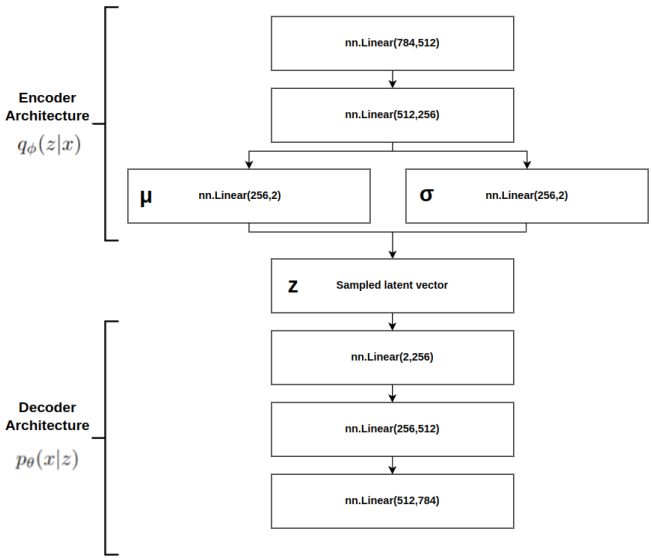


Figure 10: β-VAE Architecture used in Experiments

Notably, the final hidden layer was divided into two parts: one for the mean (μ) of the distribution and one for the standard deviation (σ). The output of the layer responsible for producing the σ vector was exponentiated to ensure that σ remains positive.

Reparameterisation: As explained in Subsection 2.2, the output of the encoder was sampled (μ and σ) to obtain a realisation from the latent distribution. We obtain the sample latent vector z as follows:

$$z = \mu + \sigma \odot \epsilon$$

where

$$\epsilon \sim \mathcal{N}(0, I)$$

Decoder Architecture: In summary, the decoder mirrors the encoder in reverse. It employs ReLU activation functions in the hidden layers and a sigmoid activation function in the output layer to produce the reconstructed vector $\hat{x}$. Specifically, the first hidden layer expands the latent vector from 2 to 256, the second hidden layer expands it from 256 to 512, and the final layer increases the size from 512 to 784.