Federated Learning under Data Heterogeneity

Nurassyl Askar and Stefano Rini ECE-NYCU, Spring 2022

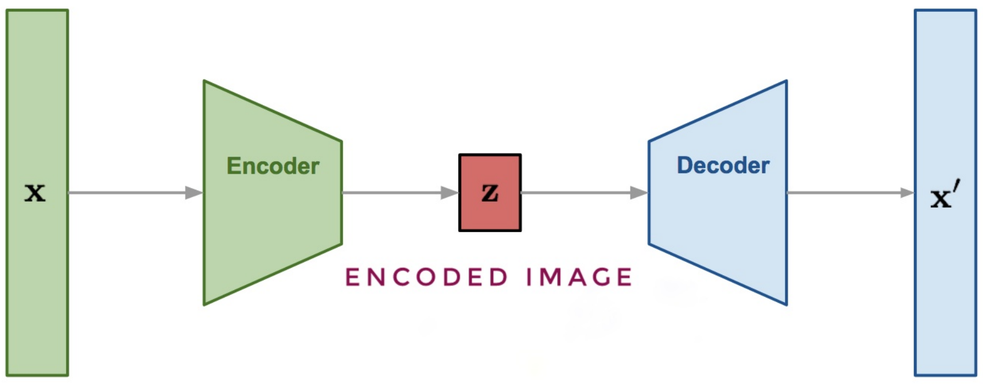
Introduction

- *Clustered Federated Learning:* Multiple users in a federated learning setting, each cluster has the same data distributition. Different data distribution across clusters.
- *Auto-encoder design:* An image is to be reconstructed by an auto-encoder under a per-pixel cross entropy loss
- *Image classes:* the images belong to a certain class – sweaters, hills, boots,... – which is a **confounding attribute** which controls the cluster data distribution

Figure 1. Fashion MNIST

- *Auto-encoder* - is a special architecture of artificial neural networks that allows the use of unsupervised learning using the back propagation method. The simplest auto-encoder architecture is a feed-forward network, without feedback, containing an input layer, a hidden layer(s), and an output layer. The output layer of an auto-encoder must contain as many neurons as the input layer.
- *Auto-encoder architecture:* Auto-encoder architecture can be very simply consisting of only one hidden layer or it can be complex with multiple layers for example convolutional auto-encoder. Current project implements simple auto-encoder architecture consisting of 5 layers. As seen from [Fig. 2], **x** is an input, **x'** is an output, and **z** is an encoded image.

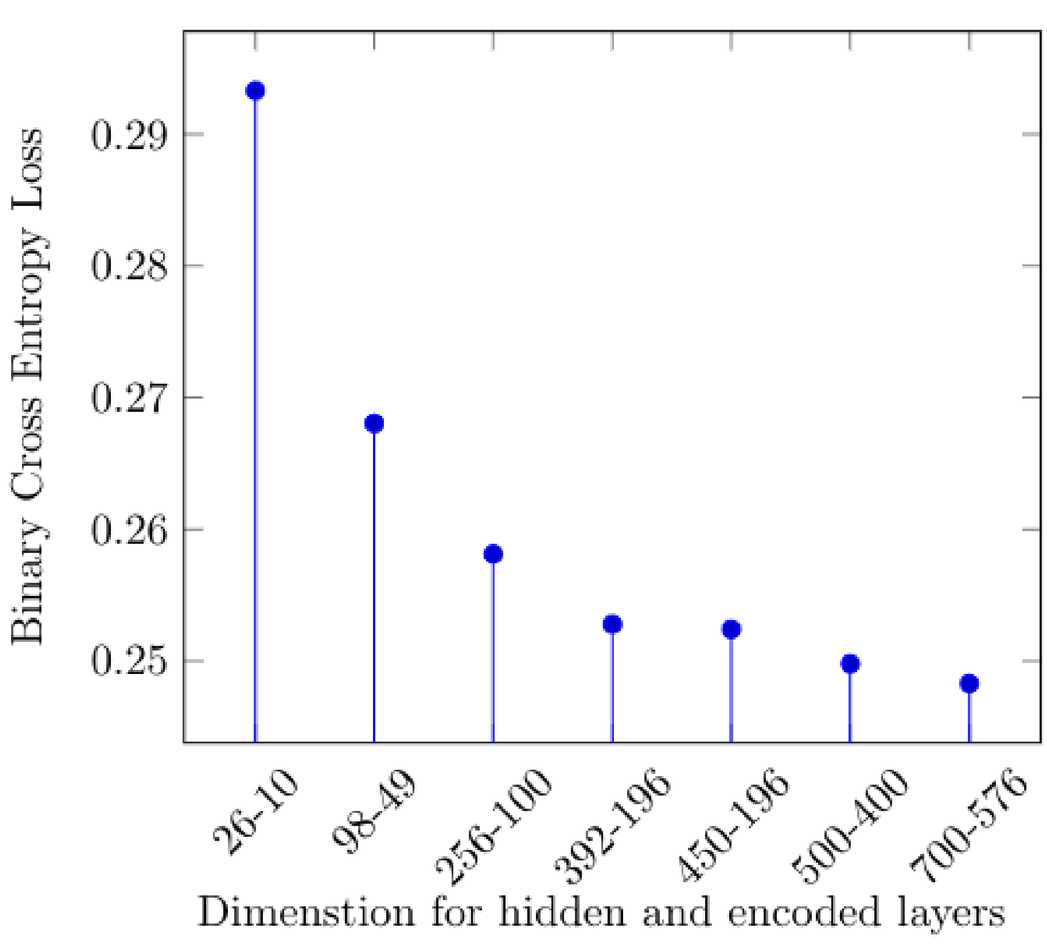
Figure 2. Auto-encoder architecture



Centralized performance

- *Purpose of Centralized Test:* In order to find best hidden layer dimensions for a given number of encoded features centralized test was performed on a whole fashion mnist dataset. The goal was determine size so that the compression is significant and a binary cross entropy loss is low.
- *Layer Dimension selection:* Input layer **x** has a shape of flattened 28x28 image which is 784. Output layer **x'** is similar to input layer. Both encoder and decoder layers consist of Dense layers. Experimenting with different encoded image sizes and sizes for hidden layers the optimal compression size was found, results shown in [Fig. 3]. These Dense layers have the following shapes 392 for the encoder and decoder layers and 196 for the encoded image layer **z**.

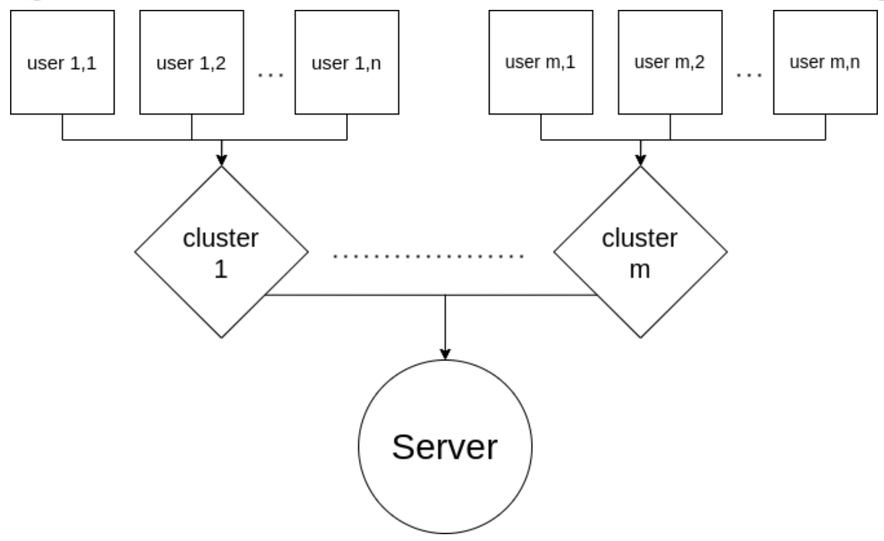
Figure 3. Centralized Performance in order to find hidden layer dimensions



Clustered Federated Learning

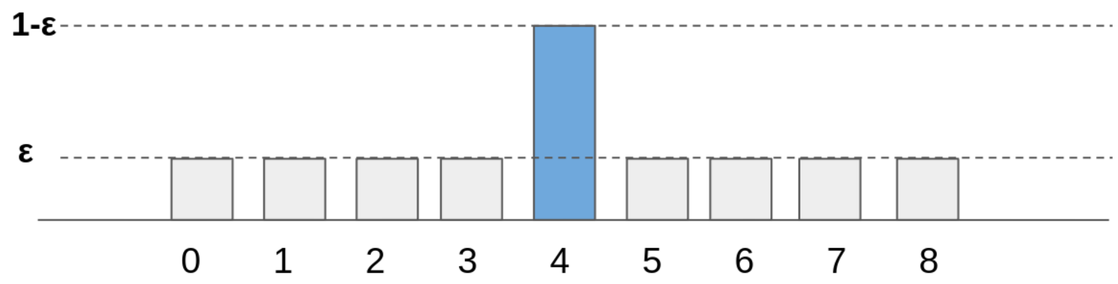
- *Server:* Centralized model that contains whole dataset for both training and for test.
- *Clusters:* 2 or more models that collect information from users and update *server* model using their own experience. Each *cluster* dataset was formed from centralized dataset, and for each of them the set of images is unique. The number of clusters used for this experiment is $m = 9$.
- *Users:* *Clusters* have multiple unique *users* that belong to them. *User* models that train the model on their own. Each *users* dataset is unique and was formed from the dataset of a *cluster* that they belong to. Total number of users used for the experiment is 45, and each cluster has $n = 5$ unique users.

Figure 4. Structure of Clustered Federated Learning



- *Clusters dataset:* Data for training and testing is biased for every *cluster*. Let ϵ represent *heterogeneity level* in range [0.1, 1.0] from 10% to 100% respectively. Portion $1 - \epsilon$ of images are from 1 of 9 labels [sweater, hills, boots, bags, dresses, pants, sneakers], and ϵ represents the remaining 8 labels [Fig. 5]. Example: 1000 images in total and $\epsilon = 0.8$, it will mean that 800 of those images are of the one type(sneakers, bags, jackets, and etc), and 200 other images are uniformly a random selected from the remaining 9 labels. Same for the test images.

Figure 5. Biasing cluster data



- *Principle of Heterogeneity level in theory:* Using data biasing the cluster can become specialized on working with specific type of data. In theory heterogeneity level should determine how specialized is a network for a certain type of data.
- *Model averaging:* Using *Model averaging method* in which each *cluster* has its own DNN model and does not mix the experience with other *clusters* during training. By averaging gradient of each *user* in *cluster* the *cluster* model is being updated. Same applies to *server* model, averaging *clusters* gradient and update *server* model.

Figure 6. Model averaging method performance

- *Training results:* From [Fig. 6] we see that in the case of $cluster_4$ which is biased towards reconstructing *Dresses*, as expected performance of a $cluster_4$ model is getting better when heterogeneity level increases. The binary cross entropy loss after 70 epochs is better for heterogeneity level 100%, then 50%, and 10% heterogeneity level showed worst results. However, in the case of $cluster_2$ which is specialized on reconstructing *Boots* showed a completely opposite results. Concluding that variative data is also a key factor.