A Benchmark of Concept Shift Impact on Federated Learning Models

Comparing the differences in performance between federated and centralized models under concept shift

Matei Ivan Tudor[1]

Supervisors: Dr. David M.J. 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: Matei Ivan Tudor
Final project course: CSE3000 Research Project
Thesis committee: Dr. David M.J. Tax, Swier Garst, Alexios Voulimeneas

Abstract

Federated learning stands as an approach to train machine learning models on data residing at multiple clients, but where data must remain private to the client it belongs to. Despite its promise, federated learning faces significant challenges, particularly when dealing with non-IID and non-stationary data. A model trained on non-stationary data can be subject to concept shift, where the data used for training faces a sudden change of concept, leading to a large performance degradation when classifying data under the new concept. This research focuses on comparing the performance of federated and centralized models under such conditions. Our objective is to evaluate the extent to which federated models are more affected by concept shift than their centralized counterparts. Through a series of experiments involving image (CIFAR-10) and tabular data (2-dimensional, linearly separable, binary-classification), we demonstrate that while federated models can achieve performances close to centralized models, they exhibit greater sensitivity to data complexity and distribution shifts. Our findings suggest that, despite centralized models being better than federated ones, the gain in performance from gathering data in one place might not outweigh the privacy concerns. Furthermore, we also find that the accuracy under concept shift is dependent on the performance on original data.

1 Introduction to Federated Learning & Concept Shift

Federated learning [1] is a machine learning framework in which the data aimed to train on is distributed across multiple clients (devices). It has predominantly seen use in the domains of IoT devices and edge computing, where the data that is generated can be of great use, but must be kept decentralized due to privacy concerns. The development and improvement of technology for such devices facilitates the implementation of federated learning, with training done on the device that is generating the data or very close to it. More recently, it has also been used in other domains where the data must also be kept private, such as the medical domain [2]. In such domains where correctly classifying data is crucial or the number of devices is high, federated learning models are presented with disadvantages comparing to centralized learning techniques. The framework encounters problems like [3][4]:

- the communication cost is high;
- not all devices are available to train at the same time;
- the data is usually heterogeneous across clients;
- the data residing at the clients is non-stationary.

The last mentioned problem is often not considered in the development of federated learning algorithms, but is commonly encountered in practical applications. If the data a model is trained on is non-stationary, a drift in concept over time in the data distribution will degrade the performance of the model. Research done in this area mostly considers the input as a stream and tries to develop an algorithm that can adapt in time to what is known as "concept drift" - the data a model was trained on has a drift in its distribution, and thus the model's accuracy on this new data will be degraded, until it adapts to the drift [5]. Concept drift refers to a change in the statistical properties of the input data distribution over time, often necessitating continuous model adaptation to maintain accuracy. In the domain of federated learning, concept drift affects the training process through clients receiving data belonging to one or more new concepts.

In contrast, we introduce the notion of "concept shift". While concept drift is a problem that involves a gradual change in the data distribution, concept shift refers to a scenario where a model is trained on its available data and then deployed in an environment where the data distribution suddenly shifts to a new, unseen concept. Whereas for concept drift the model is tasked to adapt to the incoming concept, with concept shift, the model is effectively frozen after the training process and is tasked with predicting a set of data belonging to a new concept.

This opens up the possibility to compare the performance between federated models and centralized models under concept shift. Federated models face two additional layers of complexity on top of centralized ones, that of private, decentralized data and that of non-IID data between the clients. The issue of non-IID data in federated models manifests in two ways: either data at different clients belong to different distributions, or all client data follow the same distribution but are heterogeneous in label distribution and client sample quantity (including the amount of data points per label) and in our research we will refer to non-IID data as being the latter case. These complexities motivate the need to compare federated and centralized models under concept shift, as they may cause federated models to perform worse than centralized models.

In this research, we aim to find out how much more than centralized models are federated learning models affected by concept shift. The research question tackled is the following:

Are federated models affected by concept shift more than centralized models?

This question is worth answering as we may find out that centralized models are better at learning than federated ones, and if the discrepancy in performance is high, this might signal benefits in advocating for a centralized approach. We aim to explore this question by experimenting with two types of data, image and tabular, and two classification problems: binary-classification for tabular data and multi-classification for image data. We also compare the performances of the models when training data belongs to a single concept, as opposed to having multiple concepts in the train set. To the best of our knowledge, the question proposed has not been tackled before. As such, we want to assess whether there are any considerable discrepancies even in simpler settings, without the added complexity of multiple existing concepts during training.

We start by presenting background information and by formally describing the problem in Section 2. The idea in in-

ducing concept shift for image and tabular data is described in Section 3. The setup and results of the experiments are shown in Section 4. In Section 5 we discuss choices for our experiments, other approaches, and limitations. Section 6 is comprised of the implications for responsible research. Finally, we conclude and provide guidance for future work in Section 7.

2 Background and Problem Formulation

2.1 Formal Problem Description

Shift in data is caused by many factors that change over time. Such factors can include rising trends, demographic biases, or change in users' behavior [3]. For example, the term "blockchain" saw a significant increase in usage and different search results before and after the rise of cryptocurrency in mainstream media.

Formally, given a set of samples from an unknown probability distribution $P(\mathbf{x}, y)$, where $\mathbf{x}$ is a feature vector and y is its label, a shift in the distribution corresponds to any change in the joint probability $P(\mathbf{x}, y)$. The joint probability can be factored in two ways: $P(\mathbf{x}, y) = P(\mathbf{x})P(y|\mathbf{x}) = P(y)P(\mathbf{x}|y)$. By distinguishing from the cases where $P(y|\mathbf{x})$ and $P(\mathbf{x}|y)$ are invariant, a shift corresponds to a change in one of the four probability distributions:

- $P(\mathbf{x})$ - representing a change in the input distribution, e.g. domain generalization problems;
- $P(y)$ - representing a change in the label distribution, e.g. class imbalance problems;
- $P(y|\mathbf{x})$ - representing a change in the labels of data, altering the true boundary of the model;
- $P(\mathbf{x}|y)$ - representing a change in the features of data specific to the label.

In this research we focus only on shift in $P(\mathbf{x})$ and in $P(y|\mathbf{x})$. A shift in the posterior probability $P(y|\mathbf{x})$ is also termed as *real shift*, as it alters the true boundary of the classification problem. On the other hand, a shift in the marginal probability $P(\mathbf{x})$ is termed as *virtual shift*, as only the input features are affected by the shift, but the underlying relation between them and their labels stays the same.

Concept drift can be characterized by many features such as predictability, recurrence, synchronism, as discussed in [6]. However, most of them can only be applied in the case of concept drift, and cannot be used as well for concept shift. For concept shift, only 2 characteristics can always be analyzed: form and severity. The form feature is represented by the probability distribution that is affected, for which we consider $P(\mathbf{x})$ and $P(y|\mathbf{x})$. The severity feature describes how different the new concept is from the previous one and is directly proportional to the degradation in accuracy.

2.2 Background

In the field of both centralized and federated machine learning, concept drift is a critical issue that impacts the performance and reliability of models deployed in dynamic environments. This phenomenon is well-studied in the context of data streams, where algorithms are designed to detect and adapt to these changes [7][8][9].

Research into federated learning has predominantly focused on addressing the problem of concept drift in streaming data, where the model continuously receives new data, possibly under a different concept. Several studies have highlighted the adverse effects of concept drift on model performance. For instance, Gama et al. (2014) [10] provide a comprehensive review of concept drift detection and adaptation techniques, emphasizing the need for models to be robust against such changes. Similarly, the work by Ditzler et al. (2015) [11] explores various strategies for learning in non-stationary environments, proposing methods to maintain model accuracy over time.

However, the problem of concept shift is not much discussed in the domain of federated learning. Kohli et al. (2021) perform a wide range of experimentation in a centralized environment regarding the significant performance degradation caused by distribution shifts, "where the training distribution differs from the test distribution" [12]. In their research, Kohli et al. present 2 branches of the problems caused by out-of-distribution data in the form of concept shift: domain generalization, which affects the input distribution, and class imbalance, which affects the label distribution, and they also present hybrid settings. They experiment on many datasets spanning the domains of imagery, text mining, tabular data, and even code completion, alongside much discussion of the problems mentioned above. Their results have shown that tests on out-of-distribution data can cause up to a 22% decrease in accuracy, compared to tests on in-distribution data.

As much as the settings used for concept drift adaptation may reflect the situation during training, there also comes the time where a model must actually predict data and that is where a sudden concept shift may take place. Federated learning introduces unique challenges due to its decentralized training process and the non-IID nature of client data, making it disadvantaged to centralized learning, which does not present these issues. Thus, we aim to find out how much these disadvantages affect performance of federated models under concept shift, by comparing them to a centralized model.

3 Concept Shift Simulation

To answer the question of performance comparison between federated and centralized learning techniques, we propose the following methodology. Firstly, the problem of concept drift is explored. As there is not much research done into concept shift, the different forms for this problem must be extracted from the problem of concept drift, and this was presented in our problem formalization. Secondly, research is done to identify how a shift can be induced for image and tabular data.

To simulate real shift ($P(y|\mathbf{x})$), we use a binary-classification, linearly separable problem, which has data in the form of 2-dimensional data points, represented in a tabular manner. The hyperplane acting as boundary between the two classes, in this case a line, can be rotated around its center to induce real shift. To visualize this, Figure 1 shows data points corresponding to the original dataset of such a problem. The line aiming to separate the two classes represents

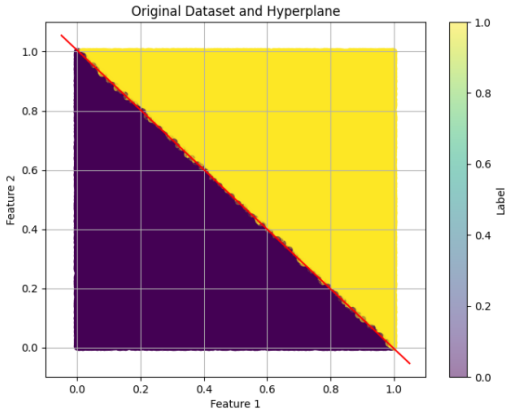


Figure 1: Original dataset and boundary

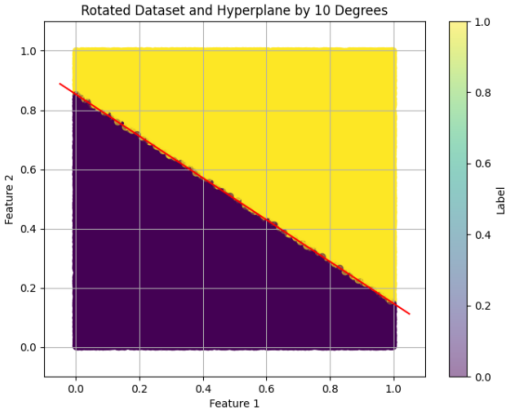


Figure 2: Rotated dataset and boundary by 10°

the boundary as was determined by a Support Vector Machine (SVM). In Figure 2, the previously found boundary is rotated around its center by 10° counterclockwise in order to induce the shift. The severity of this shift can be described by the degree of the rotation performed, with greater rotations inducing more severe shifts.

Virtual domain shift ($P(\mathbf{x})$) is simulated through the application of composition of transforms on image data. By applying transforms, we can alter the features of the input data without affecting the label. The severity feature here is related to the divergence the shifted data distribution has to the original data distribution, as well as through the cumulative variance explained by the principal components of the two distributions. The severity can also be observed through the visualization of the effects of transforms on an image, as shown in Figure 3. Figure 3a depicts the original image, Figure 3b shows a shift through increased brightness, contrast, and through grayscaling of the image. Lastly, Figure 3c presents a more severe shift by also including the addition of Gaussian noise in the composition of transforms used for the shift in Figure 3b.

(a) Original (b) Shift (c) More severe shift

Figure 3: An image before and after transforms

4 Experimental Setup and Results

For reproducibility purposes, each subsection discussing a different experiment will contain a paragraph with the specific parameters used to obtain the results. As a general remark, the core of the experiments was written making use of the PyTorch framework [13]. The federated models are using the Federated Averaging algorithm (FedAvg) [1] as it is suitable for the given tasks (further argued in Section 5). Data being distributed in a non-IID fashion is done through the usage of a Dirichlet split, with a Beta factor of 0.5. The Beta factor controls the distribution spread amongst the clients, with higher values (e.g. 1) making the spread of samples for each class between clients more even, whereas lower values (e.g. 0.1) cause a higher class imbalance. On its own, the Dirichlet split may be considered as inducing a shift in $P(y)$, as it causes class imbalance. Since we do not explore this type of shift, we choose to introduce a moderate level of heterogeneity amongst clients' class sample size to still take into account the issue of non-IID data in federated learning.

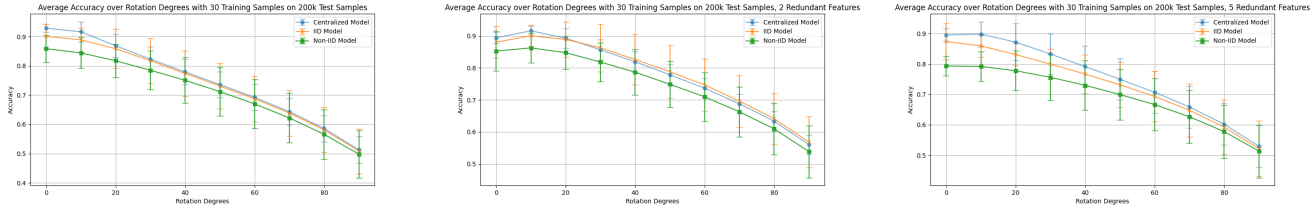
4.1 Tabular Data

For the tabular data, an artificial 2-dimensional dataset is generated using River[1], a framework that produces linearly-separable, binary-classification problems following the paper of Hulten et. al (2001) [14]. However, River does not provide the hyperplane acting as boundary between the two classes, which is needed to induce real shift as explained in Section 3. Thus, an SVM is used that extracts the hyperplane separating the two classes[2]. To induce real shift, the hyperplane is rotated around its center incrementally by 10° until 90°, where all models achieve an accuracy close to 50%. Lastly, to increase problem difficulty, we add two and five additional noisy features, with values sampled from a Gaussian distribution (mean=0, std=1). The values are then normalized to $[0, 1]$, since the values of the two features that are relevant to the class label are generated by River in this interval. These noisy features are added only with the purpose of increasing the complexity of the problem and do not affect the class boundary of the generated problem.

The problem is explored with 30, 100, and 200 samples in total. To compare performances we test the models on 2×10^5 samples. The choice for 30 samples is to simulate a low-sample size environment for learning. This also show-

[1]Link to River framework.

[2]Following this approach also implies that a number of data points will be incorrectly classified by the SVM. Fortunately, this number is significantly smaller than the total amount of data points and barely affects the overall performance.



(a) 0 noisy features (b) 2 noisy features (c) 5 noisy features

Figure 4: Accuracy with 30 training samples, 0, 2, and 5 noisy features

cases the differences between the models' learning capabilities. Secondly, we use 100 training samples to show their increased performance with a more populated dataset, although still allowing for some discrepancies in performance. Then, we test the models when having 200 training samples and five noisy features, to show how given enough samples, all models can learn the classification problem well. The problem is not explored with any more samples: the performances are related to the amount of training samples; the number of training samples required to correctly learn the problem grows as the number of features, be they redundant or not, also increases. Empirically, it was also observed that especially the non-IID model needs more training data in order to achieve a similar accuracy to the centralized and IID federated ones, meaning that the trend will be the same: increasing the number of data points will improve the performance of the models until they manage to have converge at a certain point, and adding noisy features will degrade the performance accordingly, the first noticeably affected being the non-IID federated model. A proof of convergence is not provided, however convergence is assumed through the performance when testing, having the three models trained on 10^4 samples. The observed accuracy on all models was $96\% \pm 2$.

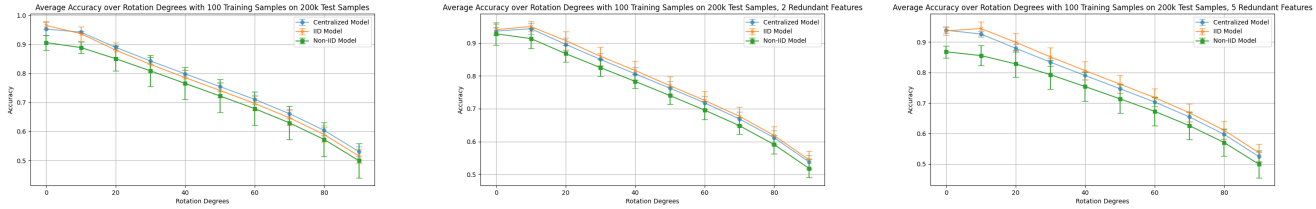
Each model is trained with an MLP with one hidden layer of 1000 nodes. The Adam optimizer is used with a learning rate of 10^{-3}. The centralized model is trained for 50 epochs. The federated models are trained for 20 rounds, have data split between 10 clients, and 2 randomly chosen clients are trained each round. Each client trains its own copy of the global model for 5 epochs. To run the experiments, each model was trained 5 times, and an average of their performances was recorded. This was done due to the inconsistencies in accuracy for the non-IID federated model caused by the Dirichlet split, as well as the model being inherently stochastic in the initialization stage. Training is done solely on non-rotated data. The original datasets are obtained by using seed two when running the hyperplane generator and by providing the number of data points to generate.

Figure 4 shows how a small number of data points affects the learning of the three models. As it is empirically observed from the subplots, the non-IID federated model performs worse than the other two. Although this is not by much in the case of 0 and 2 noisy features, when 5 noisy features are introduced, a significant difference in performances can be seen, of 10% between the non-IID model and centralized model. In this case, the IID federated model is also more affected by the noise than the centralized model, although it still achieves noticeably higher accuracies than its non-IID counterpart. This large discrepancy between the models is due to the small number of data points. Nevertheless, as they are further tested on increasingly rotated data, the discrepancy between the models decreases, until it converges for all close to 50% accuracy.

In Figure 5, the number of data points is increased to 100. For 0 and 2 noisy features, the difference between performances shrinks. The accuracy of all three models is increased compared to the previous settings, although the non-IID model still underperforms in contrast to the other two models. Adding 5 noisy features still affects the non-IID model very much. On the other hand, the IID model is now also not at all affected by the noisy features, indicating that it is more capable of differentiating the redundant features than the non-IID model.

One aspect worth noting is presented in Figure 5b, where the initial performance is almost the same for all three models, but the non-IID model performs slightly worse upon rotating the boundary. The issue causing this performance difference is related to the boundary the model is predicting. The boundary found may differ from the true one and can lead either to increased accuracy upon first few rotations, or to decreased accuracy overall. For example, considering the true boundary is the one in Figure 1, the former case would take place if the predicted boundary is closer to the boundary in Figure 2, rather than the true one. The latter case would take place if the boundary found is rotated in a clockwise direction, relative to the true one.

With 200 samples, there are already enough data points such that each model can easily learn the classification, even with noise. Furthermore, the problem of finding a boundary close to the true one is also much diminished now. Figure 6 shows the accuracy of all three models with 200 training samples and 5 noisy features. The discrepancy between the performance of the non-IID federated model and the other two is lower now, only 5%. The IID federated and centralized models achieve an almost identical initial accuracy and continue to have such a performance for the rotations as well. Even though the non-IID model starts with a lower accuracy, the performance over rotations quickly catches up to the other two models' accuracy.



(a) 0 noisy features (b) 2 noisy features (c) 5 noisy features

Figure 5: Accuracy with 100 training samples, 0, 2, and 5 noisy features

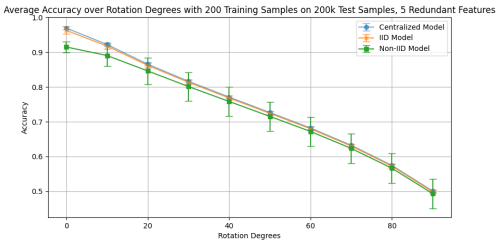


Figure 6: Accuracy with 200 training samples, 5 noisy features

4.2 Image Data

To explore domain shift, we make use of the CIFAR-10 dataset [15], as it is provided by the PyTorch framework. We evaluate performances under concept shift with both only original training data and when 20% of the clients have transformed, in-distribution data. In the latter approach, the centralized model will have the same transformed data as the non-IID federated model had. The IID federated model will also have 20% of the clients with transformed data, however this data might be different than for the other two. We consider this transformed data to be in-distribution, rather than being out-of-distribution and representing a shift of concept compared to the original data.

We induce domain shift by applying transforms on the test data, such as color jitter, full image grayscaling, horizontal flips, Gaussian blur, and Gaussian noise. We perform five different experiments regarding the transforms applied, and we present them in Table 1. In our first experiment ("None"), no transforms are applied to the training set. In our second experiment, "Train", we present transforms we consider to generate in-distribution data and are comprised of random image grayscaling, taking place with a 50% chance, and color jitter, with a brightness parameter of 0.5 and a contrast parameter of 0.4. A brightness/contrast parameter of n represents a change in the brightness/contrast of the image by a factor chosen uniformly in $[\max(0, 1-n), 1+n]$. For example, a brightness factor of 0.5 yields an image that is half as bright as the original, and the same goes for contrast. The third experiment ("Aggressive") is comprised of full image grayscaling (100% chance), more aggressive color jitter, with a brightness parameter of 1 and a contrast parameter of 0.9, and random horizontal flips taking place with a 50% chance. In experiment four, we add the Gaussian blur transform to the composition of "Aggressive" transforms, with the kernel being 3 pixels wide and high, and the standard deviation being chosen uniformly in $[0.1, 2]$. Lastly, in experiment five we add a Gaussian noise transform (mean = 0, std = 0.1) to the composition of "Aggressive" transforms.

Such transforms are commonly used in data augmentation techniques [16] to create data coming from a distribution similar to the original one. There is no precise measure to determine whether transformed data represents an augment or a concept shift from the original. However, some transforms produce distributions closer to the original, while more aggressive transforms create distributions that diverge more significantly. For our purposes, we consider the composition of transforms that minimally diverges from the original distribution as generating in-distribution data. Conversely, compositions of more aggressive transforms are assumed to cause a concept shift, resulting in out-of-distribution data.

However, rather than discussing the benefits of data augmentation, we are trying to find out how well the models can also learn the features from their two clients' augmented data, as we expect the centralized model to do better. We do this to simulate a situation where the new concept has features in common with parts of the training data, and the models actually benefit from having clients with the "Train" transformed data, even though the train and test sets differ. For example, when testing on the new concept represented by the "Aggressive" transform, common features are provided by the images that are grayscaled during the "Train" transform. Furthermore, the chosen intervals for the brightness ($[0.5, 1.5]$) and contrast ($[0.6, 1.4]$) factors are both subsets of the intervals used for the "Aggressive" transforms ($[0, 2]$ for brightness and $[0.1, 1.9]$ for contrast).

To back up our previous assumptions, in Table 1 we present the Jensen-Shannon (JS) divergence [17] between the feature distributions of the transformed and original data. To extract the features, we use a ResNet-50 [18] with pre-trained weights, as is provided by PyTorch. As stated before, the "Train" transform is used in the case of 20% clients with altered data. It produces data closest to the original distribution, with a JS divergence of 0.06, and its purpose is to prepare the models for the concept shift. The "Aggressive" transforms and its blur and noise counterparts are used to test the performance of the models under concept shift. The "Aggressive" transform has a larger divergence to the original data, with a JS divergence of 0.14. The addition of the Gaussian blur transform further increases this divergence to 0.17, while the

addition of the Gaussian noise transform shows a 0.19 divergence.

Another measure we look at is the dissimilarity in the cumulative variance explained of the principal components of the original and transformed data. In Table 1 we denote "C m" as the cumulative variance explained for the first m components, and we calculate this value for each of the first three, 10^{th}, and 100^{th} components. We do this by applying Principal Component Analysis on the training dataset (after applying transforms on it), and then we calculate the cumulative sum for each of these component sets. The most notable change is observed in the first principal component. While there is a 0.13 increase in the captured variance of the first component in the "Train" transformed data compared to the original data, the first component of the "Aggressive" transformed data already accounts for 0.36 more variance than that of the original data. An increase in the first component means that now a larger part of the variance can be explained by a single component. We also base our assumptions on this increase that the "Train" transform does not present a new concept, whereas the "Aggressive" transforms do.

Continuing, adding Gaussian blur to the transforms, the first component accounts for 0.05 more variance than that of the "Aggressive" transform did. When adding Gaussian noise to the transforms, the variance becomes more distributed throughout the components, although there still is a single component accounting for a large part of the variance. This variance distribution can be seen as for the "Train" transform, the variance starts at 0.42 for the first component and reaches 0.92 with 100 components, while for the "Aggressive & Noise" transform, it starts higher at 0.60 but only reaches 0.91 with 100 components. However, with the "Aggressive & Noise" transform, the JS divergence is again increased. We expect this to lead to an even lower performance when testing on this transform, compared to the other two aggressive transforms.

Each model is trained with a ResNet-50[3] [18] with pretrained weights, as provided by the PyTorch framework. The fully-connected layers are replaced with a new classifier with two other fully-connected layers with 1024, respectively 512 nodes. The Adam optimizer is used with a learning rate of 2×10^{-5}. The centralized model is trained for 40 epochs. The federated models are trained for 20 rounds, have data split between 10 clients, and all clients are selected for training each round, as the classification task is difficult. Training with all clients at once also stabilizes very much the training process, ensuring that different runs only have insignificant variations. Each client trains its local model for five epochs. These numbers of epochs and rounds were chosen because around these training iterations, test accuracy gains became more erratic, and train accuracy hardly saw an increase anymore. The models are then tested in the case of only original data and of eight clients with original data, two clients with "Train" transformed data.

[3]By using a ResNet, after applying transforms and converting images to tensors for training, we also normalize values to the ImageNet mean and standard deviation values and scale the train and test images to 224×224, in order to increase classification performance.

We now discuss the accuracy of the models in each of these settings. In Table 2 it can be seen that the centralized model with altered data performs better than all other models in all settings except in the case of original test data, although the difference is only 0.2%. There is a clear difference in the performance achieved by the centralized model and the ones achieved by the federated models. As expected, the federated models are disadvantaged comparing to the centralized model, with the non-IID model being somewhat weaker than its IID counterpart but still achieving almost the same accuracy. We can also observe that when testing against the "Aggressive" transformed data, the gap between the centralized model with transformed clients and any of the 2 federated models is largest. This goes to show that the centralized model can better make use of the altered data than the federated models, which backs our previous motif for using the "Train" transform on parts of the training data.

When testing on "Aggressive & Blur" or "Aggressive & Noise", even though the blur and noise transforms do not provide a much larger increase in divergence, the impact on accuracy can clearly be seen. As these shifts increase in severity, we also observe that the benefits of the transforms on the two clients' data is much smaller now.

However, when looking at the full picture, the centralized model is not better by much. With "Train" transforms in the initial data, the maximum observed increase in any setting between the centralized model and any of the two federated ones was of 5% accuracy, whereas without the transforms, the maximum gap was of 3.2%. This goes to show how well the federated approach can work compared to the centralized one, given its multiple disadvantages.

5 Discussion

In this section, we present an analysis of our current approach, how other approaches might yield different results, and the limitations of our experiments.

5.1 Training with all clients at once

While for the tabular data experiment we used 20% of clients to train in a round, for image data we used all clients. We do this as the the classification task for CIFAR-10 is more difficult than the one with tabular data.

Initially, experiments for CIFAR-10 were run as well using only 20% clients per round. However, we observed a large performance decrease with the non-IID model compared to the IID one. We show the performance of both federated models, with and without 20% clients having "Train" transformed data in Table 3, and we test their accuracy on the original and "Train" transformed test set. The decrease in accuracy of the non-IID federated model is further observed under concept shift in Table 4, where we denote "0_NIID" as the model with original train data, and "20_NIID" as the model with 20% "Train" transformed clients. Given these results, we wanted to find out whether the possibility to train with all clients at once improves the non-IID model's performance. We did the same with the IID model to see if it could match the accuracy of the centralized model. However, as shown in Table 2, only the non-IID model managed to have a significant increase in performance, very close to the one of the

Transforms	JS Divergence	C 1	C 2	C 3	C 10	C 100
None	0	0.29	0.40	0.47	0.65	0.90
Train	0.06	0.42	0.52	0.58	0.73	0.92
Aggressive	0.14	0.65	0.71	0.75	0.84	0.95
Aggressive & Blur	0.17	0.70	0.76	0.80	0.88	0.98
Aggressive & Noise	0.19	0.60	0.66	0.70	0.78	0.91

Table 1: Differences in the transforms used compared to original data

Model, Transformed Clients	No Tr.	Train Tr.	Aggressive Tr.	Aggressive & Blur Tr.	Aggressive & Noise Tr.
Centralized, 0%	**84.02%**	69.94%	44.57%	28.58%	18.94%
Centralized, 20%	83.80%	**72.79%**	**51.00%**	**30.16%**	**19.40%**
IID Federated, 0%	82.20%	67.49%	42.89%	26.47%	17.16%
IID Federated, 20%	82.05%	69.61%	46.74%	29.09%	16.94%
n-IID Federated, 0%	79.81%	66.05%	43.18%	26.51%	16.30%
n-IID Federated, 20%	81.33%	69.02%	45.31%	28.39%	16.76%

Table 2: Classification performance of each model on test data under different transforms

Model, Transformed Clients	No Tr.	Train Tr.
IID Federated, 0%	81.70%	66.50%
IID Federated, 20%	81.59%	67.43%
non-IID Federated, 0%	74.24%	57.83%
non-IID Federated, 20%	73.81%	58.91%

Table 3: Federated models' classification performance on CIFAR-10 when training with only 2 clients per round

Model	Aggressive	Ag.& Blur	Ag. & Noise
0_NIID	34.51%	21.85%	15.14%
20_NIID	37.88%	22.61%	15.38%

Table 4: Classification performance of non-IID model on CIFAR-10 under concept shift with only 2 clients per round

IID model. A gap still remained between the accuracy of the centralized and federated models. This is due to the issue of decentralized data, as the averaging done by FedAvg is less precise than the calculations of the centralized model.

5.2 On the usage of only FedAvg

Federated Averaging [1] is probably the most basic federated algorithm, and was a pillar in the following development of other algorithms. However, in the case of concept drift, many papers describe it as being obsolete, usually achieving very bad performance in comparison to their approaches, which detect and adapt to the concept drift.

Research into concept drift usually also takes into account issues such as allowing the model to train only once on the data, as it is received, as well as training with data comprised of multiple concepts at once. Moreover, as the problem of concept drift implies continuous client data updates, training is also done while a new concept is emerging throughout the clients' data, and adaptation to the concept is of paramount importance. However, there is no issue in our settings that puts the FedAvg algorithm at disadvantage, as we only train on in-distribution data, with clients' data being unchanged throughout the training stage, and test on data belonging to a new concept.

If our experiments were to also include such considerations, then FedAvg would most certainly fail. For example, training a model under multiple concepts requires a more sophisticated approach than simply averaging local model weights, such as maintaining a different global model for each existing concept [19]. Nevertheless, with this approach comes one matter to take into consideration: if a drift-specific algorithm is used by the federated models, then the same drift-specific algorithm must be adapted and utilised by the centralized model, in order to achieve comparable performances.

5.3 Limitations

Limitations were encountered in the way shift was induced for image data and in the performance of the classifier on the CIFAR-10 test set.

Domain shift can be induced through transforms, domain generalization, or purely through existing data that can be divided into drifted and non-drifted. However, for the last two approaches, we had no such information from the CIFAR-10 dataset. As such, we experimented only with transforms.

The ability of a classifier to learn and correctly predict data also has a large impact on the overall performance of a model. For example, the centralized model was only able to achieve 84% accuracy on the original test set, as was presented in Table 2. From this we can expect the federated models to achieve an accuracy no higher than this one. State-of-the-art models for CIFAR-10 could provide better performances, both in centralized and federated settings and might even lower the gap between them. However, such models are very complex, and their reproduction is hard, and we deemed this approach out of scope.

6 Responsible Research

Upon performing research, responsibility is a key aspect to take into consideration. Reproducibility of experiments, transparency of presented results, as well as their credibility,

are all part of a responsibly conducted research. In this study, we have taken account of these matters through careful planning and execution of experiments. Moreover, this research has gone through two stages of peer review, with valuable feedback having been incorporated in this work.

To further enhance reproducibility and ensure transparency, for both experiments we provide the exact configurations used that lead to our results. Moreover, the code for the experiments has been uploaded to a public GitHub repository[4], and instructions to run the experiments are in the "README.md" file. For the tabular experiments, we also provide the original artificially generated datasets, as well as the rotated datasets for easier use. As such, exact reproduction of both experiments is either available by running the code in the repository, or by following the steps in the Experimental Setup and Results section. We note however that, while rather insignificantly, results might still slightly vary: the initialization of machine learning models is stochastic and federated clients receive different data upon a different run. The transforms performed in the image data experiment also differ in choice of parameters from run to run (e.g. the same image's brightness might be modified differently upon a new run than it was previously), however this is out of control as it is internally performed by the PyTorch framework, and multiple runs were performed to ensure results did not present a significant difference.

We also provide for both experiments the necessary time to train the three models. For the image data experiment, training the centralized model takes 02:30 hours, whereas training the federated models takes 03:30 hours. For the tabular data experiment, training takes 10-15 seconds for all three models. However, the hardware used for the experiments significantly influences training times. All experiments were conducted on a laptop with an RTX 4060 and 24GB of RAM, with computations done on the GPU. Variations in hardware configurations, especially performing computations on a CPU rather than on the GPU, may lead to differences in execution times.

7 Conclusions and Future Work

7.1 Conclusions

In conclusion, our initial question can be answered affirmatively: federated models can indeed be more affected than centralized models. However, the extent of this difference in performance depends on the complexity of the problem, the classifier's ability to learn, as well as the performance of the model on the original test data.

Our current results show that federated learning's disadvantages are evident even in simpler experiments with tabular data. Nevertheless, with a sufficient number of data points, federated models can catch up to centralized models, achieving similar performance levels. In the context of our tabular data experiments, federated models can perform as well as centralized ones under real concept shift, although they are more prone to accuracy drops as the problem becomes harder to learn through the introduction of noisy features.

For more complex problems, such as image classification with CIFAR-10, training with all clients simultaneously significantly enhanced federated models' performance. Although federated models appeared to be less powerful than centralized models, they could still learn effectively, with only a slight performance gap. In this experiment, the centralized model better utilized the 20% of clients with transformed, in-distribution data when the new concept shared features with this data, but the improvement was minor (2.5%) and quickly outweighed by the addition of blur and noise transforms.

Another conclusion is that non-IID data poses a greater challenge than data decentralization. In our tabular data experiments, the IID model achieved performance levels comparable to the centralized model, while the non-IID model required more data points to better match their accuracy. Furthermore, as discussed in Section 5.1, this issue becomes more pronounced as the classification task increases in difficulty, such as in our image data experiments. Although we improved the performance of the non-IID model by training with all clients simultaneously, almost matching the performance of the IID model, there was little to no improvement for the IID model. It is important to note that despite training with all clients, a performance gap remained between the centralized and federated models. This gap reflects the impact of decentralized data, as FedAvg struggles to match the centralized model's computations. However, the performance decrease due to decentralization is significantly less severe than that caused by non-IID data.

Moreover, we also note that the performance under concept shift is also dependent on the performance on the original data, as was shown in Table 3 and Table 4. This was also seen in the tabular data experiment, where the model with non-IID data would have a lower initial accuracy than the IID and centralized model. It would continue to have a lower performance overall, even if the gap in accuracy between the non-IID model and other two decreased, as the rotation degree increased.

Finally, we found that, as the severity of the shift increases, the accuracy gap between the three models decreases, which was observed in both image and tabular data experiments.

Overall, while federated models can achieve accuracy levels close to centralized models, their performance is more susceptible to being affected. They do not significantly lag behind but struggle to generalize as well as centralized models and are more sensitive to problem complexity and learning environment. Furthermore, increased problem complexity (such as CIFAR-10 classification) highlights the benefits of training with all clients simultaneously, most observed when the federated model has clients with non-IID data. However, despite the centralized model's higher accuracy in both experiments, the overall gap did not exceed 10%, which might not justify the privacy trade-offs of centralizing data.

7.2 Future Work

Further work is comprised of experimentation with different types of problems, other types of data to work with, and other datasets to use. For example, one can make use of textual data, which on its own brings new problems (e.g. text classification). Image segmentation, node and graph classification, when tabular data can be represented in the form of graphs,

[4]Link to concept shift GitHub repository.

are also different problems to be explored with image and tabular data. In [12], Kohli et. al present a number of datasets already having distribution shifts that do not cause any performance drops, however, as noted previously, they only discuss experiments in a centralized manner, and this is not yet known for federated models as well. Lastly, class overlap would be an interesting issue to experiment with, as it makes the problem of learning fundamentally harder.

In the context of our experiments on tabular data, there is also future work to be done. Specifically, the problem should be explored when the boundary between classes is non-linear, as such problems are somewhat harder to learn than linear ones. Additionally, it would also be necessary to explore the problem in multi-classification, rather than just binary-classification. For both of these problems, we refer the reader to the THU Concept Drift Datasets[5] [20], a framework capable of generating linear and multiple types of non-linear problems of concept drift, with many ways of simulating real shift. However, the framework is made for data streams, so there will be the need of some adaptability to the problem of concept shift.

References

[1] H. Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Agüera y Arcas. Communication-efficient learning of deep networks from decentralized data, 2023.

[2] Seongjun Yang, Hyeonji Hwang, Daeyoung Kim, Radhika Dua, Jong-Yeup Kim, Eunho Yang, and Edward Choi. Towards the Practical Utility of Federated Learning in the Medical Domain, May 2023. arXiv:2207.03075 [cs].

[3] Peter Kairouz, H. Brendan McMahan, Brendan Avent, Aurélien Bellet, Mehdi Bennis, Arjun Nitin Bhagoji, Kallista Bonawitz, Zachary Charles, Graham Cormode, Rachel Cummings, Rafael G. L. D'Oliveira, Hubert Eichner, Salim El Rouayheb, David Evans, Josh Gardner, Zachary Garrett, Adrià Gascón, Badih Ghazi, Phillip B. Gibbons, Marco Gruteser, Zaid Harchaoui, Chaoyang He, Lie He, Zhouyuan Huo, Ben Hutchinson, Justin Hsu, Martin Jaggi, Tara Javidi, Gauri Joshi, Mikhail Khodak, Jakub Konečný, Aleksandra Korolova, Farinaz Koushanfar, Sanmi Koyejo, Tancrède Lepoint, Yang Liu, Prateek Mittal, Mehryar Mohri, Richard Nock, Ayfer Özgür, Rasmus Pagh, Mariana Raykova, Hang Qi, Daniel Ramage, Ramesh Raskar, Dawn Song, Weikang Song, Sebastian U. Stich, Ziteng Sun, Ananda Theertha Suresh, Florian Tramèr, Praneeth Vepakomma, Jianyu Wang, Li Xiong, Zheng Xu, Qiang Yang, Felix X. Yu, Han Yu, and Sen Zhao. Advances and Open Problems in Federated Learning, March 2021. arXiv:1912.04977 [cs, stat].

[4] Yujing Chen, Zheng Chai, Yue Cheng, and Huzefa Rangwala. Asynchronous Federated Learning for Sensor Data with Concept Drift, August 2021. arXiv:2109.00151 [cs].

[5] Link to THU concept drift GitHub repository

[5] Gustavo Oliveira, Leandro Minku, and Adriano Oliveira. Tackling Virtual and Real Concept Drifts: An Adaptive Gaussian Mixture Model, February 2021. arXiv:2102.05983 [cs].

[6] G. Yang, X. Chen, T. Zhang, S. Wang, and Y. Yang. An impact study of concept drift in federated learning. In *2023 IEEE International Conference on Data Mining (ICDM)*, pages 1457–1462, Los Alamitos, CA, USA, dec 2023. IEEE Computer Society.

[7] YiMin Wen, Xiang Liu, and Hang Yu. Adaptive tree-like neural network: Overcoming catastrophic forgetting to classify streaming data with concept drifts. *Knowledge-Based Systems*, 293:111636, June 2024.

[8] Giuseppe Canonaco, Alex Bergamasco, Alessio Mongelluzzo, and Manuel Roveri. Adaptive Federated Learning in Presence of Concept Drift. In *2021 International Joint Conference on Neural Networks (IJCNN)*, pages 1–7, Shenzhen, China, July 2021. IEEE.

[9] Yingying Chen and Hong-Liang Dai. Concept drift adaptation with continuous kernel learning. *Information Sciences*, 670, 2024.

[10] João Gama, Indrė Žliobaitė, Albert Bifet, Mykola Pechenizkiy, and Abdelhamid Bouchachia. A survey on concept drift adaptation. *ACM Computing Surveys*, 46(4):1–37, 2014.

[11] Gregory Ditzler, Manuel Roveri, Cesare Alippi, and Robi Polikar. Learning in Nonstationary Environments: A Survey. *IEEE Computational Intelligence Magazine*, 10(4):12–25, November 2015.

[12] Pang Wei Koh, Shiori Sagawa, Henrik Marklund, Sang Michael Xie, Marvin Zhang, Akshay Balsubramani, Weihua Hu, Michihiro Yasunaga, Richard Lanas Phillips, Irena Gao, Tony Lee, Etienne David, Ian Stavness, Wei Guo, Berton A. Earnshaw, Imran S. Haque, Sara Beery, Jure Leskovec, Anshul Kundaje, Emma Pierson, Sergey Levine, Chelsea Finn, and Percy Liang. Wilds: A benchmark of in-the-wild distribution shifts, 2021.

[13] Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. Automatic differentiation in pytorch. 2017.

[14] Geoff Hulten, Laurie Spencer, and Pedro Domingos. Mining time-changing data streams. In *Proceedings of the seventh ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 97–106, San Francisco California, August 2001. ACM.

[15] Alex Krizhevsky. Learning Multiple Layers of Features from Tiny Images. 2009.

[16] Connor Shorten and Taghi M. Khoshgoftaar. A survey on image data augmentation for deep learning. *Journal of Big Data*, 6(1), July 2019.

[17] J. Lin. Divergence measures based on the shannon entropy. *IEEE Transactions on Information Theory*, 37(1):145–151, 1991.

[18] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep Residual Learning for Image Recognition, December 2015. arXiv:1512.03385 [cs].

[19] Ellango Jothimurugesan, Kevin Hsieh, Jianyu Wang, Gauri Joshi, and Phillip B Gibbons. Federated Learning under Distributed Concept Drift. 2023.

[20] Zeyi Liu, Songqiao Hu, and Xiao He. Real-time safety assessment of dynamic systems in non-stationary environments: A review of methods and techniques. In *2023 CAA Symposium on Fault Detection, Supervision and Safety for Technical Processes (SAFEPROCESS)*, pages 1–6, 2023.