1. Introduction
2. Background

2.1 Machine Learning in Bioinformatic fields

What can be done using ML (several examples using Neural network), may other ways that use old machine learning.

2.2 Performance and dataset

If there’s a paper about data size and training performance… Introduce several famous datasets in biology/medical fields. But compare to large dataset/baseline like ImageNet xxx. Or describe the need for dataset like lack sth???

2.3 Federated Learning

What is / why use FedML (privacy/distribute computing or storing resources), challenges, federated learning frameworks. Federated success case in biology field.

1. Datasets and Methods

In this project, the CIFAR-10 dataset has been used for locally simulating different scenarios of federated learning, the OIA-DDR dataset has been used for implementing an example medical image classification task on the cloud. The performance of the models will not be high enough for any actual usage, only for testing federated learning in this project.

We test 1000 epochs (for centralized training) or global rounds (for federated learning) for each simulation task and 100 epochs/rounds for the cloud implementation. Our code is written in Python3 and TensorFlow 2.0.1 is used.

3.1 CIFAR-10 dataset

The CIFAR-10 dataset consists of 50,000 training images and 10,000 testing images of 10 balanced classes, each image is in color format and the size is 32x32 pixels.

3.2 OIA-DDR dataset

The OIA-DDR dataset provides high-quality diabetic retinopathy (DR) images and annotations. The grading annotation labels the images to six classes: no DR, mild, moderate, severe, proliferative, and ungradable. In this project, we neglect images from the ‘ungradable’ class, using 6266 images from ‘no DR’ class as healthy samples and 6256 images from the rest 4 classes as DR samples.

Images in this dataset are in different sizes and qualities. We preprocess the images using the methods from public kernels on Kaggle, crop and resize them to 224x224 pixels. (<https://www.kaggle.com/ratthachat/aptos-eye-preprocessing-in-diabetic-retinopathy>) (<https://www.kaggle.com/titericz/circle-to-rectagle-preprocessing-1>)



(Figure xxx each class’s count)

3.3 Training designs

The task for CIFAR-10 is to input images of size 32x32x3 and classify them to 10 classes. The model is provided by Mattias Åkesson. We apply ADAM optimizer with learning rate set to 1e-3. The batch size is set between 50 to 100; loss will be calculated by categorical cross-entropy.

The task for OIA-DDR dataset is to input images of size 224x224x3 and classify them to 2 classes. VGG16 network (pre-trained on ImageNet dataset) is used. We use ADAM optimizer with learning rate initialized at 1e-4 and decayed by rate 0.05 in every epoch. The batch size is set to 300; loss will be calculated by binary cross-entropy.

learning\_rate = 1e-4/(1+0.05\*epoch\_number)

3.4 Scenarios testing

We simulate and test different scenarios in federated learning in this part. Due to the limit of time and computational resources, each test is performed only once.

***Simulation System***

One central node and several worker nodes are involved in this simulation system. We assume no data or nodes will be dropped out from or added in to the system during the entire training process.

When the system starts, the central node will initialize a model. We apply Federated Averaging in this system, weighted by node’s size. Each global round consists of the following basic actions:

1. Worker nodes load weight from the central node,
2. Worker nodes train the model over their local data and save the history,
3. Worker nodes return weights to the central nodes,
4. Central node aggregates the weights and evaluates the model.

After simulation, we collect the evaluation results in central node’s folder as the overall testing history and average the node’s training history as the overall training history.

***Pilot Tests***

Before the formal implementation of different scenarios, we first test the system with different sizes of dataset and different non-IID classes. Figure xxx compares centralized and federated learning’s behavior with different size of training dataset. The result fluctuates due to randomness, but it is obvious that the performance of both federated and centralized learning can be improved with larger training set, and federated learning always converge at slower speed than the centralized learning.

Figure xxx

The figure shows the training result with different size of dataset. (window size=21) LEFT: This plot shows the windowed accuracy at minimum windowed loss. RIGHT: This plot shows the global rounds to reach to minimum windowed loss.

In previous works we already know that a non-IID training set can result in performance degradation in federated learning. There is no obvious standard to quantify the degree of non-IID, so we show it by identifying how many classes each node has. Figure xxx shows federated learning’s behavior at different non-IID levels. It is obvious that accuracy and speed will decrease as the degree of non-IID increases.

(Figure xxx. Pilot test)

LEFT: This figure shows the windowed accuracy of the global round with minimum windowed loss for different non-IID classes of the first 500 rounds. RIGHT: This figure shows the global rounds to reach to minimum windowed loss for different non-IID classes of the first 500 rounds (window size=21).

We also have observed different trends in accuracy between tests using dataset of size 4000 and 40000 under 2 class non-IID circumstances.

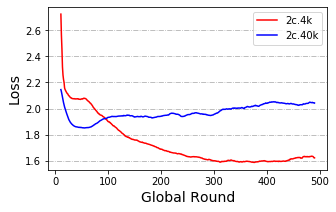


Figure xxx

4k v.s. 40k dataset under 2 classes non-IID condition.

***Default Simulation Settings***

We will select a balanced total training set of size 4000 or 40000 and apply evaluation with Cifar10’s whole testing set. Training set are distributed to 10 or 40 nodes depending on the experimental design. Worker nodes are in equal size and samples from available classes within a node are balanced. By default, local epochs are set to 1 and global rounds are set to 1000. Evaluation is performed within the central node. In each scenario, we change 1 parameter and compare it to the one without that change.

we test each scenario under 2 classes, 5 classes and IID classes circumstances with dataset of size 4000 (4k dataset) and under 2 classes, 3 classes, 5 classes and IID classes circumstances with dataset of size 40000 (40k dataset). Early stop of 500 rounds is applied to testing with 4k dataset.

***Model Poisoning***

We test this scenario where there is an abnormal node that returns arbitrary weights at each global round. We split the training set to 10 nodes and added in an abnormal node who claims to be 1% or 10% as large as total training dataset.

Testing is performed with 4k and 40k dataset. Additional tests of 3% abnormal node size are performed with 40k dataset. Other addition tests of poisoning frequency and poisoning data are performed with 40k dataset under IID condition.

***Data Dispersion***

In some tests we distribute our data to 10 workers, in other tests to 40 workers. In this scenario, we compare the training performance between 10 worker and 40 worker cases.

***Delayed Update***

In this scenario, some nodes update at slower speed than others. We split the training set to 40 nodes and test the situation from 2 aspects:

1. Different proportion of delayed nodes: 25%, 50% or 75% nodes delayed, they load the weights at global round n, and upload the weights at global round n+3. New weights won’t be loaded until current weights are uploaded.
2. Different speed of delayed nodes: 20% nodes delayed, they load the weights at global round n, and upload the weights at global round n+3, n+12 or n+30. New weights won’t be loaded until current weights are uploaded.

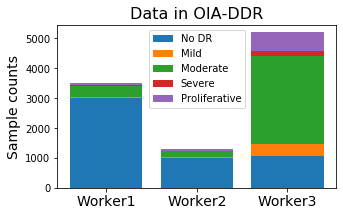
***Share data strategy***

This scenario is tested with 1 class, 2 classes, 5 classes and IID classes, we split the training set to 10 nodes. In pilot tests, 1 class non-IID never behaves better than random. A previous research suggested a data-sharing strategy that can improve the training performance over 1 class non-IID data (Zhao *et al.* 2018). The author assumes a small IID set (2.5% to 25% as large as total training dataset) can be published for pre-training the model or sharing between nodes.

We apply this strategy to our model with sharing size equals to 2.5% of the total training set. In each node there will be a mixture of shared data and local data. In our design, an IID local training set is randomly resampled from the local sample pool. We compare the result of our design to the original strategy. In order to simplify the pipelines, pre-training step is not applied.

3.5 Cloud implementation (NOT yet finished)

This



1. Results

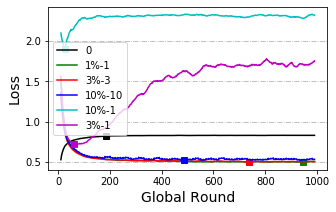
4.1 Scenarios testing

In this section we will show summaries of our testing results, the full results are available in the appendix. In an ideal training, maximum accuracy and minimal loss will be achieved at converge epoch. In our trainings, we have observed slight increase in accuracy when the loss reaches to its least, in which case the model already overfits the training data (training accuracy reaches to 1). Thus, we assume the model converge at the global round with minimal loss, the accuracy at that round represents model’s best performance.

Since the original results are fluctuated, we will show and manipulate on windowed testing accuracy and loss in this report, with window size equals to 21. The full and original records and are available in the GitHub repository.

***Model Poisoning***

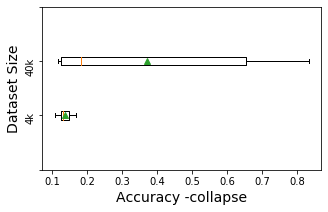
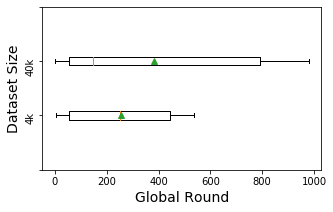
The presence of abnormal node leads to performance degradation. The influence is related to both the size of the abnormal node and the poisoning frequency. We label the situation when an abnormal node is as large as x% of the total training set and upload arbitrary weight by every y round as ‘x%-y’. We observe similar accuracy trends among situation ‘1%-1’, ‘3%-3’ and ‘10%-10’, while there is a huge difference among ‘1%-1’ , ‘3%-1’ and ‘10%-1’. In the 3 similar situations, they have the same average abnormal node size per round. The influence of size of abnormal node in this case still exists. Loss of ‘10%-10’ start increasing at earlier rounds than ’1%-1’.

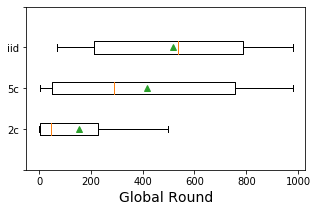
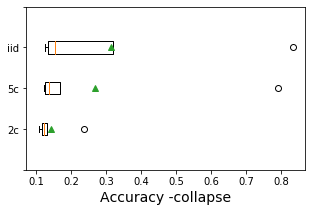


Large and IID dataset can increase model’s tolerance to the attack from abnormal nodes. In the case of 1% size abnormal node, we cannot observe performance degradation in model trained with an IID 40k dataset, while this degradation is obvious in models trained with smaller or non-IID dataset (see appendix).

In figure xxx and figure xxx below, we show the speed of collapsing by the index of first collapse round and the severity of performance decrease by the accuracy of that round (using ‘x%-1’ results only). We identify the performance to be collapsed when the accuracy decreased to its lowest point after the minimal loss (a threshold of 0.025 was applied to eliminate the influence from fluctuation). If accuracy does dot collapse within 1000 global rounds, collapse index is set as 1000.

We assume training with large or IID dataset can collapse at slower speed than with small or non-IID dataset, and its accuracy at collapse point will also be higher. Though we can see obvious patterns in their means, variance within each testing groups is large and not consistent. Paired T-test has been applied between each IID groups or dataset size (exclude ‘3%’ results when comparing between dataset sizes). Unfortunately, there is not statistical significance between IID levels or dataset sizes.



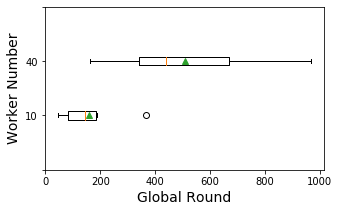
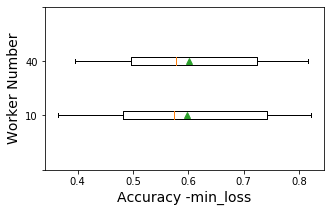
(in dataset size comparison, ‘3%’ is neglected)

We also calculate Pearson correlation coefficient between model’s best performance (accuracy at minimal loss) and collapse speed (index of collapse round). The result shows a positive correlation (0.92, p-value = 1.33e-06). Positive correlation also appears between model’s best performance and the accuracy at collapse round (0.70, p-value = 0.0035). It means seriously affected trainings or less robust models will collapse at faster speed and to worse results. 在0.3 acc时候check的频率应该高于0.8 （discussion里）

Testing results in data poisoning indicates that data poisoning is weaker than model poisoning.

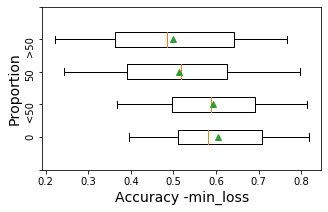


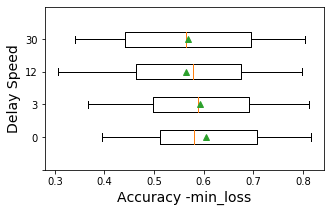
***Data Dispersion***

From Figure xxx we can see training with 40 worker nodes is obviously slower than training with 10 worker nodes while the convergence accuracy is not obviously affected. Paired T-test is applied and the result shows a significance in speed (p-value=0.0059). Differences in accuracy at convergence is not significant (p-value=0.71).

***Delayed Update***



Paired T-test and ANOVA is applied to show the difference between no-delay node and the other situations. The ANOVA test shows no significance in both accuracy and speed in different groups by proportion or by delay speed. On the other hand, T-test shows a significance between with or without delayed update in convergence speed when delay speed is 3 and porportion less than 50 (p-value=0.016). Surprisingly, this significance disappears as delay speed and porportion increases. Also, convergence accuracy with delayed porportion larger than 50 is reported to be significantly reduced (p-value=0.01 and 0.005). Delay speed does not obviously affect convergence accuracy.

***Share data***

In this scenario we compare different methods to use the shared data. As we mentioned before, each worker owns a mixed pool of local and shared data. When the shared is small, data in the pool is still unbalanced.

Figure xxx shows the results between directly use the mixed dataset and do resample on the mixed dataset when sharing data is small (as large as 2.5% of total dataset). The direct method performs better than the resample method in most cases except for in 1 class non-IID scenario. In such scenario, accuracy of direct method decreases sharply after overfitting while accuracy of the resample method can stay stable.

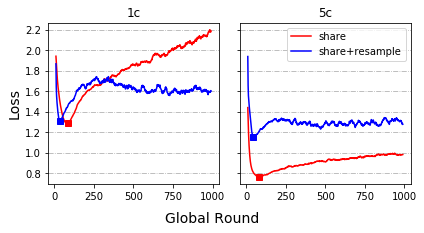
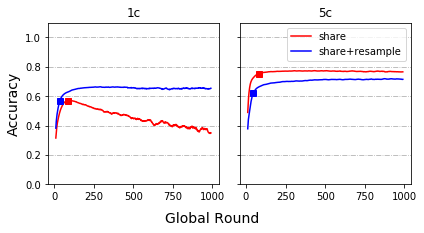
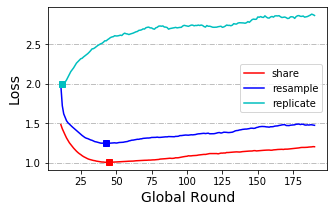
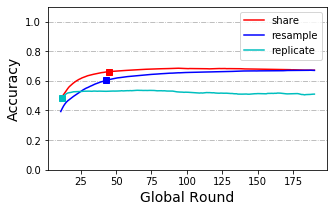
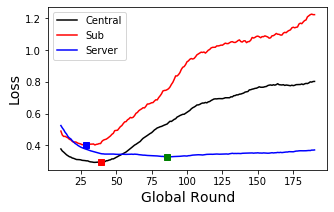


Figure xxx shows the result of replicating the small classes in the pool to make labels balanced. This method performs worst.



4.2 Cloud implementation

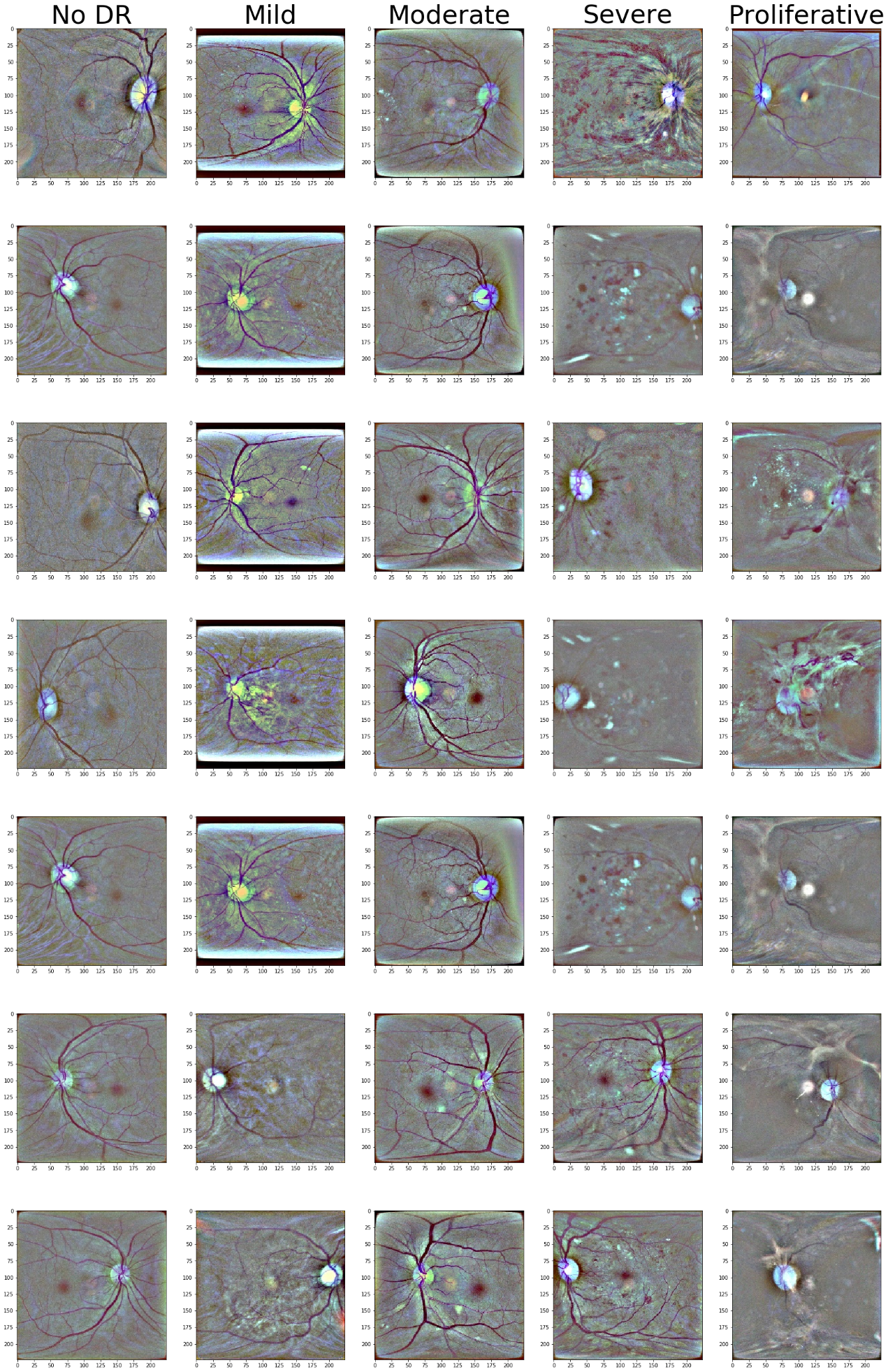


1. Discussion

(like how to evaluate on node.)

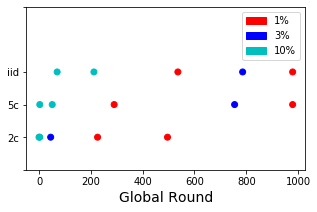
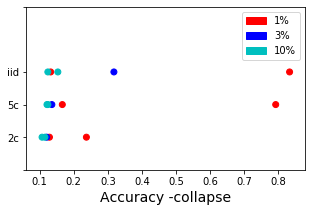
1. Conclusion
2. Acknowledgement
3. Reference

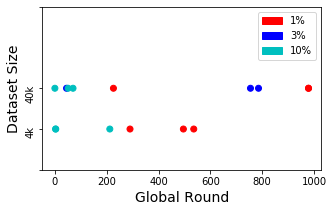
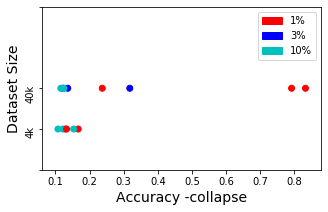
Appendix 1 OIA-DDR dataset after preprocessing

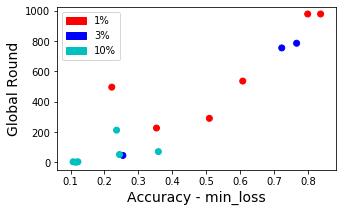


Appendix – Dot plots

***Model Poisoning***







***Data Dispersion***



***Delayed Update***





Appendix –p-value records

***Model Poisoning***

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | ‘4k’-‘40k’ | ‘IID’-‘5c’ | ‘5c’-‘2c’ | ‘iid’-‘2c’ | ANOVA |
| Acc\_collapse | 0.15 | 0.30 | 0.30 | 0.20 | 0.54 |
| Round\_collapse | 0.43 | 0.25 | 0.13 | 0.085 | 0.29 |

***Data Dispersion***

|  |  |
| --- | --- |
|  | ‘10w’-‘40w’ |
| ACC | 0.71 |
| Round | 0.0059 \* |

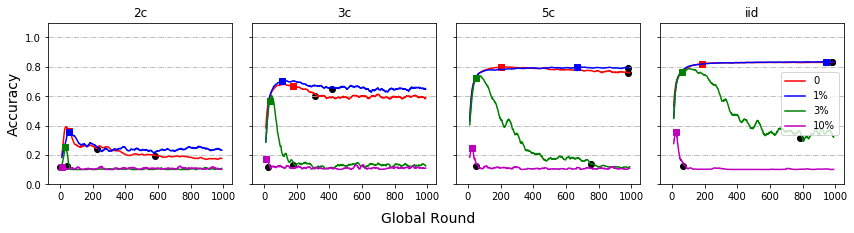
***Delayed Update***

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Proportion | ‘0’-‘<50’ | ‘<50’-‘50’ | ‘50’-‘>50’ | ‘0’-‘50’ | ‘0’-‘>50’ | ANOVA |
| Acc | 0.41 | 0.0041 \* | 0.39 | 0.01 \* | 0.005 \* | 0.62 |
| Round | 0.016 \* | 0.90 | 0.60 | 0.089 | 0.092 | 0.35 |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Delay Speed | ‘0’-‘3’ | ‘3’-‘12’ | ‘12’-‘30’ | ‘0’-‘12’ | ‘0’-‘30’ | ANOVA |
| Accuracy | 0.41 | 0.027 \* | 0.77 | 0.11 | 0.052 | 0.97 |
| Round | 0.016 \* | 0.014 \* | 0.44 | 0.17 | 0.059 | 0.23 |

Appendix- Scenario testing results with 40k training set

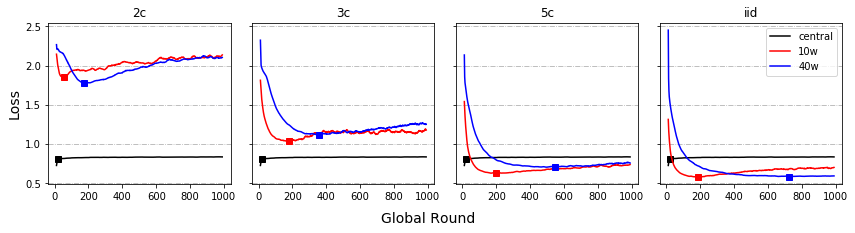
***Model Poisoning***





***Data Dispersion***





***Delayed Update – By Proportion***

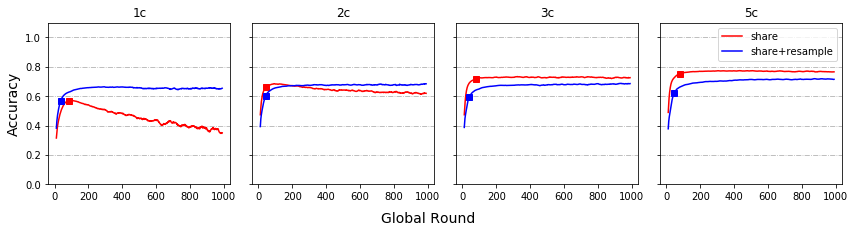


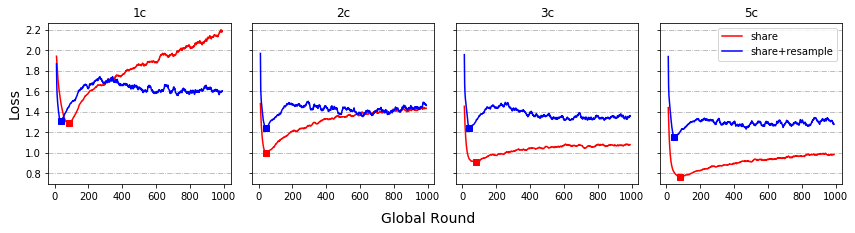
***Delayed Update – By Speed’***





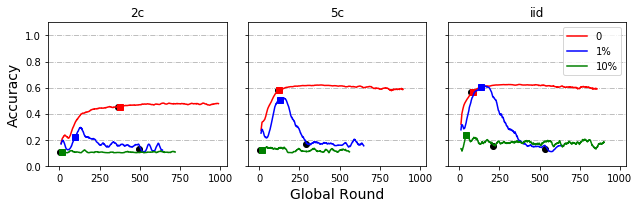
***Share data***





Appendix- Scenario testing results with 4k training set

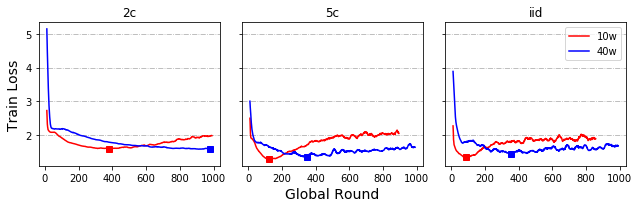
***Model Poisoning***





***Data Dispersion***





***Local Round***





***Delayed Update – By Proportion***





***Delayed Update – By Speed***





***Share data***

