**Abstract:** Federated Learning (FL) enables distributed training on multiple devices, enhancing privacy and conserving resources by sharing model updates instead of data. Using NVFlare, we distributed the training of a CNN for brain tumor detection, keeping sensitive data local. We evaluated different FL algorithms (FedAvg, FedOpt, FedProx, Scaffold) and found that complex algorithms like Scaffold perform better among heterogeneous data distributions.

**Introduction**

**Traditional Machine Learning Approach:**
- Train a model such that it recognizes a pattern or behavior.
- Labeled data (supervised learning) or unlabeled data (unsupervised learning) needed.
- Data is centralized in one spot.

**Problem:**
What if data cannot be shared and allocated in one spot due to privacy, data restrictions or resource limitations?

**Federated Learning:**
An approach enabling multiple peers to collaboratively learn a shared prediction model by sharing the weights of the model but not the data itself.

**Research Objective**

**Research Statement:**
Traditional distributed learning assumes:
- Data on different nodes (computing units) originate from the same distribution.
- Data on different nodes have a similar size.

Real-world scenarios often feature:
- Significant data imbalances.
- Size differences in data across nodes.

**Federated Learning method:**
Tailored to handle unbalanced and non-identically distributed (non-IID) data using different Aggregation Algorithms.

**Key Algorithms in FL:**
- FedAvg:
  - Collects and aggregates local weights using a weighted average after local training.
- FedProx:
  - Adds a loss function to penalize local weights deviating from the global model.
- FedOpt:
  - Allows use of a specified optimizer and learning rate scheduler (e.g., SGD) to aggregate model weights.
- Scaffold:
  - Adds a correction term to neural network parameters during local training by calculating the discrepancy between global parameters.

**OBJECTIVE:**

**FL Methodology**

**FL Framework:**
NVIDIA FLARE from NVIDIA
- Allows adapting an existing machine learning workflow to a federated paradigm.

**Data splitting methods For FL:**
Various splitting methods were used to simulate balanced and imbalanced data distributions among the sites.
- Uniform: No imbalance, each site has the same amount of data.
- Square: The amount of data is correlated with the site-ID in a squared fashion ($1^2$ to $ID^2$).
- Exponential: The amount of data is correlated with the site-ID in an exponential fashion ($exp(1)$ to $exp(ID)$).

**Hyperparameters for experiments:**
- Distribution of data per site varies from 5% to 50% (of the training data).
- Batch size of 32, 15 epochs, Learning rate of 0.001.
- 5 rounds of training (Receive model, train locally and share updates).
- 5, 7 Clients.

**Experiment Tracking:**
- Usage of the Mlflow instance provided by AI4EOSC Project
- Tracking of loss and accuracy for training and validation and other parameters.

**Results**

**Global Model Performance for 5 clients**

**Results**

**Global Model Performance for 7 clients**

**Discussion:**
- The federated version of the machine learning workflow can keep up with the base line depending on the number of clients in combination with the right algorithm and data distribution.
- Accuracy with FedAvg gets worse the more unequal the distribution of the data.
- Scaffold performs better for heterogeneous data.
- For a small count of clients the algorithms had just a marginal difference.

**Outlooks**

- Compare different FL libraries and evaluate them (privacy, platform compatibility, versatility, ...).
- Investigating the FL methods in more complex task such as object detection and segmentation.
- Test the shown use case on real world sites and compare the results to the simulation (speed, overhead, accuracy, ...).
- Further investigation into different FL algorithms and development of new approaches.

**References**