Universidade Federal
do Espírito Santo

CENTRO TECNOLÓGICO
DEPARTAMENTO DE INFORMÁTICA

Sistemas Distribuídos – 2023/1
Prof. Rodolfo da Silva Villaça – rodolfo.villaca@ufes.br
Monitor: Eduardo M. Moraes Sarmento – eduardo.sarmento@ufes.br
Trabalho T1 – Implementação de Aprendizado Federado

Objetivos:

1. Implementar o aprendizado federado;
2. Sincronizar a troca de mensagens entre os componentes do sistema;
3. Analisar os resultados da resolução de problemas utilizando aprendizado de máquina.

Definições Gerais:

1. O trabalho pode ser feito em grupos de 2 ou 3 alunos: não serão aceitos trabalhos individuais ou em grupos de mais de 3 alunos;
2. O trabalho deve ser implementado usando a linguagem Python;
3. O problema a ser resolvido é o de classificação de imagens. Para isso usaremos o *dataset* MNIST, o mesmo utilizado no Laboratório 2;
4. Para a resolução do problema usaremos a mesma arquitetura de rede neural usada no Laboratório 2;
5. O trabalho deverá possuir um servidor de agregação disponível em rede, com endereço IP e porta disponíveis para acesso pelos clientes (treinadores). Ex: 127.0.0.1:8080.

Requisitos Gerais:

1. O *dataset* deve ser dividido entre os clientes de treinamento. A divisão fica a critério de cada grupo;
2. O paradigma de aprendizado federado a ser implementado é o centralizado, ou seja, existe um servidor de agregação e clientes que se conectam a ele;
3. O servidor de agregação deverá ser inicializado com o número de clientes a serem escolhidos em cada *round* de treinamento, quantidade mínima de clientes participando em cada *round*, quantidade máxima de rounds necessários para concluir o treinamento, meta de acurácia

(usada para parar o processo caso já tenha sido alcançada antes do final do processo) e *timeout* de conexão com os clientes;
4. Deverão ser inicializados pelo menos 3 clientes, executando em processos diferentes, e conectados ao servidor por meio do seu endereço IP;
5. Os clientes devem, a cada *round*, enviar um ID único de identificação do cliente, o endereço IP do cliente com a porta, em contrapartida, o servidor deve enviar aos clientes o número do round atual no início de cada rodada de treinamento (*round*);
6. O servidor deve esperar o número mínimo de clientes se conectarem naquele *round* para que ele escolha, de maneira aleatória, os clientes que irão fazer parte do conjunto de treinamento daquele *round*;
7. Os clientes escolhidos devem treinar seus modelos usando os dados locais. Cada cliente que terminar de treinar, deve enviar os pesos do seu modelo local para o servidor;
8. O servidor então deve esperar todos os clientes enviarem seus pesos e agregá-los por meio do algoritmo *Federated Average* (*FedAvg*), a definição do algoritmo se encontra no primeiro artigo da bibliografia;
9. Por último, o servidor deve enviar aos clientes os pesos agregados. Os clientes irão atribuir os pesos agregados aos seus modelos locais, e farão a avaliação do modelo, por meio de métricas, usando seus dados locais. Os resultados atingidos após essa atualização devem ser impressos na tela de cada cliente e enviados ao servidor, onde eles serão agregados e a métrica agregada é comparada a meta de acurácia, concluindo, assim, um *round* do processo de aprendizado federado;

O processo descrito nos Itens 4 a 8 devem se repetir até que se atinja a quantidade máxima de *rounds* ou a meta de acurácia, sendo que esses parâmetros são atribuídos ao servidor em sua inicialização. Ao final do processo deve ser gerado gráficos mostrando a evolução do valor das métricas de acurácia de cada cliente durante os rounds de treinamento.

Requisitos Específicos:

- Alunos de Graduação – O meio de conexão entre o servidor e os clientes deve ser implementado a partir do protocolo gRPC

(apresentado no Laboratório 3). Para isso devem ser implementadas as seguintes chamadas de procedimento remoto:

1. Chamada de registro feita pelos clientes ao servidor passando o IP, a porta, e o ID único do cliente. O servidor deve retornar um código de confirmação ao cliente e o número do round atual;
2. Chamada de início do treinamento, feita pelo servidor aos clientes escolhidos. Os clientes devem retornar os pesos encontrados no treinamento e o número de amostras da base de dados local.
3. Chamada de avaliação do modelo feita do servidor aos clientes, mesmo para os que não foram escolhidos para treinamento, passando os pesos agregados de cada round. Os clientes devem retornar o resultado das métricas de avaliação (acurácia, neste exemplo).

- Alunos de Pós-Graduação – O meio de comunicação entre o servidor e os clientes deve ser implementado comunicação indireta por meio de *middleware Publish/Subscribe* com Filas de Mensagens (apresentado no Laboratório 4). O trabalho deve ser interoperável entre grupos, isto é, o servidor de um grupo deve ser capaz de se comunicar com o cliente de outro grupo e vice-versa. É necessário a implementação das seguintes mensagens:

  1. Mensagem de registro publicada pelos clientes para o servidor passando o ID único do cliente. Como resposta, o servidor deve publicar mensagem para o cliente se ele foi escolhido para participar daquele *round* (ou não) e o número do *round* atual;
  2. Mensagem de agregação publicada pelos clientes ao servidor enviando os pesos do modelo local e a quantidade de amostras da base de dados local usadas no treinamento. O servidor deve publicar mensagem com os pesos agregados para todos os clientes registrado, mesmo para aqueles que não foram escolhidos para o processo de treinamento;
  3. Mensagem de avaliação, publicada por todos os clientes, mesmo aqueles que não foram escolhidos para o processo de treinamento, para o servidor passando os resultados das métricas

encontradas. Caso a meta de acurácia tenha sido atingida, o servidor publicar mensagem para os clientes indicando a parada do processo de treinamento.

Entrega:

1. Por meio da Sala de Aula Virtual da disciplina no *Google Classroom*, na atividade correspondente ao Trabalho I. 1 (uma) submissão por grupo é suficiente;
2. Deve-se submeter apenas o *link* para o repositório virtual da atividade (Github, Bitbucket, ou similares) contendo: i) códigos-fonte; ii) instruções para compilação e execução; iii) relatório técnico (*.pdf* ou *markdown, README.md*); e iv) vídeo curto (máximo 3 min) mostrando uma execução de exemplo, resultado e análise da execução;
3. O relatório técnico deverá conter: a metodologia de implementação e testes usada, resultados apresentados sob a forma gráfica, e análise e avaliação dos resultados (Ex: o resultado esperado foi alcançado? Comente!);
4. Avaliação: Adequação aos Requisitos (30%), Legibilidade do Código (30%), Documentação (40%);
5. Data de Entrega: 21/05/2023;

Bom trabalho!

Bibliografia:
[1] Definição formal de Federated learning e Federated averaging:
Communication-Efficient Learning of Deep Networks from Decentralized Data
[2] Python grpc:
https://grpc.io/docs/languages/python/basics/
[3] Broker de mensagens emqx:
https://www.emqx.io/docs/en/v3.0/
[4] MNIST Dataset:
https://www.kaggle.com/datasets/oddrationale/mnist-in-csv