



Federated Deep Learning For Edge Computing

Creation of a federated learning platform for emulation

Student: Ian See Soong En

Supervisor: Asst Prof Tan Rui

```

Node: h3
', 'self', (), {}, []
Connection was closed without reason, Server is still open
{<Task finished coro=<WebsocketServerWorker._consumer_handler() done, defined at
/home/mininet/anaconda3/lib/python3.7/site-packages/syft/workers/websocket_serv
er.py:78> result=None>}
{<Task pending coro=<WebsocketServerWorker._producer_handler() running at /home/
mininet/anaconda3/lib/python3.7/site-packages/syft/workers/websocket_server.py:1
12> wait_for=<Future pending cb=[<TaskMakeupMethWrapper object at 0x7f1eb0ec0078
>()]>>}
Connection open
<Task pending coro=<WebsocketServerWorker._consumer_handler() running at /home/m
ininet/anaconda3/lib/python3.7/site-packages/syft/workers/websocket_server.py:78
>>
<Task pending coro=<WebsocketServerWorker._producer_handler() running at /home/m
ininet/anaconda3/lib/python3.7/site-packages/syft/workers/websocket_server.py:10
0>>
worker <WebsocketServerWorker id:h3 #objects:0> received CMD [['send_central', '
self', (), {}, []]
worker <WebsocketServerWorker id:h3 #objects:0> received CMD [['stop_monitoring'
, 'self', (), {}, []]
worker <WebsocketServerWorker id:h3 #objects:0> received CMD [['dataset', 'self'
, (), {}, []]
Dataset

```

Worker Nodes

```

Node: h1
Drop outs:0
time_elapsed:14.281353235244751
h10 has dataset of 56358
Starting round0
{'h2': 0, 'h3': 0, 'h4': 0, 'h5': 0, 'h6': 0, 'h7': 0, 'h8': 0, 'h9': 0, 'h10':
0}
[('h2', 0), ('h3', 0), ('h4', 0), ('h5', 0), ('h6', 0), ('h7', 0), ('h8', 0), ('
h9', 0), ('h10', 0)]
['h2', 'h3', 'h4', 'h5']
worker <VirtualWorker id:me #objects:0> sending (OBJ <ObjectWrapper id:571217210
15 obj:TracedModule[Net](
(conv1): TracedModule[Conv2d](
(conv2): TracedModule[Conv2d](
(fc1): TracedModule[Linear](
(fc2): TracedModule[Linear](
))> to <WebsocketClientWorker id:h2 #objects local:0 #objects remote: 0>
worker <VirtualWorker id:me #objects:0> sending (OBJ <ObjectWrapper id:335395788
47 obj:ScriptModule()) to <WebsocketClientWorker id:h2 #objects local:0 #object
s remote: 2>
worker <VirtualWorker id:me #objects:0> sending (OBJ <TrainConfig id:91860311225
owner:me epochs: 1 batch_size: 1000 optimizer_args: {'lr': 0.01}>) to <Websocke
tClientWorker id:h2 #objects local:0 #objects remote: 2>
Training round 0, calling fit on worker: h2

```

Central Node

```

Node: h4
', 'self', (), {}, []
Connection was closed without reason, Server is still open
{<Task finished coro=<WebsocketServerWorker._consumer_handler() done, defined at
/home/mininet/anaconda3/lib/python3.7/site-packages/syft/workers/websocket_serv
er.py:78> result=None>}
{<Task pending coro=<WebsocketServerWorker._producer_handler() running at /home/
mininet/anaconda3/lib/python3.7/site-packages/syft/workers/websocket_server.py:1
12> wait_for=<Future pending cb=[<TaskMakeupMethWrapper object at 0x7f7282f2f048
>()]>>}
Connection open
<Task pending coro=<WebsocketServerWorker._consumer_handler() running at /home/m
ininet/anaconda3/lib/python3.7/site-packages/syft/workers/websocket_server.py:78
>>
<Task pending coro=<WebsocketServerWorker._producer_handler() running at /home/m
ininet/anaconda3/lib/python3.7/site-packages/syft/workers/websocket_server.py:10
0>>
worker <WebsocketServerWorker id:h4 #objects:0> received CMD [['send_central', '
self', (), {}, []]
worker <WebsocketServerWorker id:h4 #objects:0> received CMD [['stop_monitoring'
, 'self', (), {}, []]
worker <WebsocketServerWorker id:h4 #objects:0> received CMD [['dataset', 'self'
, (), {}, []]
Dataset

```

Worker Nodes

```

Node: h2
', 'self', (), {}, []
worker <WebsocketServerWorker id:h2 #objects:0> received OBJ <ObjectWrapper id:5
7121721015 obj:ScriptModule(
(conv1): ScriptModule()
(conv2): ScriptModule()
(fc1): ScriptModule()
(fc2): ScriptModule()
)>
worker <WebsocketServerWorker id:h2 #objects:1> received CMD [['objects_count',
'self', (), {}, []]
worker <WebsocketServerWorker id:h2 #objects:1> received OBJ <ObjectWrapper id:3
3539578847 obj:ScriptModule()>
worker <WebsocketServerWorker id:h2 #objects:2> received CMD [['objects_count',
'self', (), {}, []]
worker <WebsocketServerWorker id:h2 #objects:2> received OBJ <TrainConfig id:918
60311225 owner:h2 epochs: 1 batch_size: 1000 optimizer_args: {'lr': 0.01}>
worker <WebsocketServerWorker id:h2 #objects:2> received CMD [['fit', 'self', ()
, {'dataset_key': 'targeted'}], [0]]
/home/mininet/anaconda3/lib/python3.7/site-packages/syft/frameworks/torch/hook/h
ook.py:795: UserWarning: To copy construct from a tensor, it is recommended to u
se sourceTensor.clone().detach() or sourceTensor.clone().detach().requires_grad_
(True), rather than torch.tensor(sourceTensor).
current_tensor = hook_self.torch.native_tensor(*args, **kwargs)

```

Worker Nodes

```

Node: h5
', 'self', (), {}, []
Connection was closed without reason, Server is still open
{<Task finished coro=<WebsocketServerWorker._consumer_handler() done, defined at
/home/mininet/anaconda3/lib/python3.7/site-packages/syft/workers/websocket_serv
er.py:78> result=None>}
{<Task pending coro=<WebsocketServerWorker._producer_handler() running at /home/
mininet/anaconda3/lib/python3.7/site-packages/syft/workers/websocket_server.py:1
12> wait_for=<Future pending cb=[<TaskMakeupMethWrapper object at 0x7f943d21b078
>()]>>}
Connection open
<Task pending coro=<WebsocketServerWorker._consumer_handler() running at /home/m
ininet/anaconda3/lib/python3.7/site-packages/syft/workers/websocket_server.py:78
>>
<Task pending coro=<WebsocketServerWorker._producer_handler() running at /home/m
ininet/anaconda3/lib/python3.7/site-packages/syft/workers/websocket_server.py:10
0>>
worker <WebsocketServerWorker id:h5 #objects:0> received CMD [['send_central', '
self', (), {}, []]
worker <WebsocketServerWorker id:h5 #objects:0> received CMD [['stop_monitoring'
, 'self', (), {}, []]
worker <WebsocketServerWorker id:h5 #objects:0> received CMD [['dataset', 'self'
, (), {}, []]
Dataset

```

Worker Nodes

Project Objectives:

This project is the first part of a combined project. The aim is to create an efficient federated learning scheduling algorithm to reduce the cost of communication while maintaining model accuracy and low variance. In order to test this algorithm, an infrastructure that supports the network emulation and federated learning must be created.

Using Mininet virtual image and the PySyft federated learning library, a virtual image was created that supports testing of algorithms with the use of multiple nodes and different network capabilities