

---

# Training and Deployment in CrypTen

Anmol Agarwal

2018CS10327

Under the supervision of Prof. Rijurekha Sen



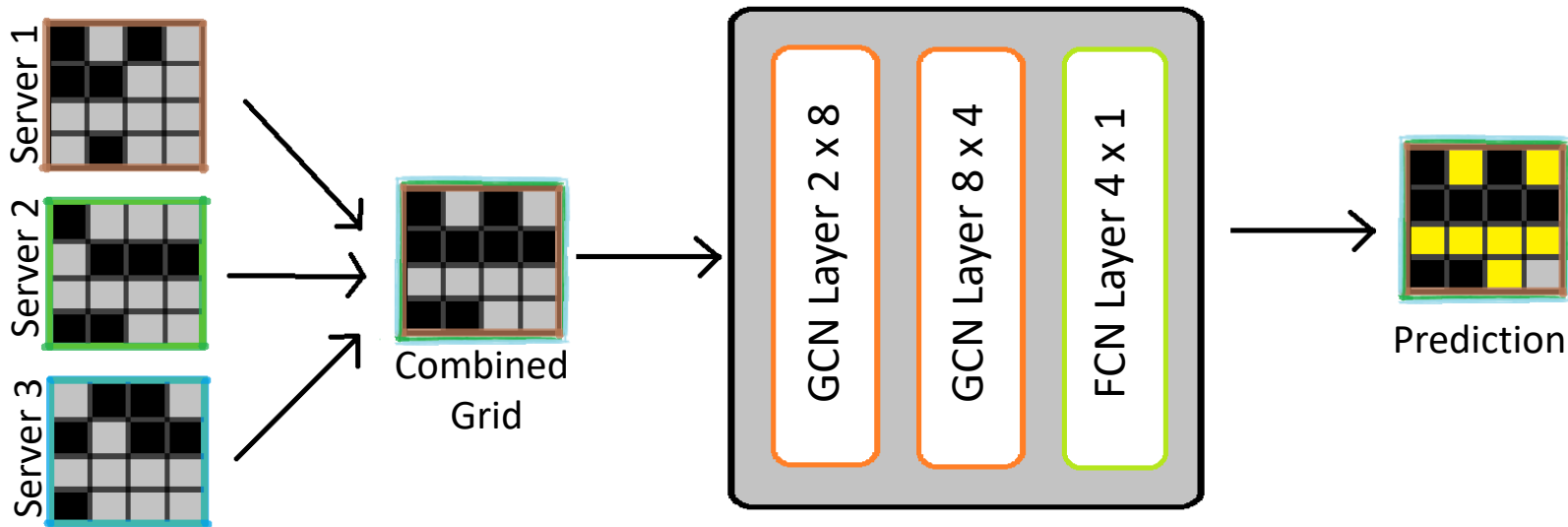
# Content

---

- Problem Description
- Secret Sharing Protocols
- Porting to newer version of CryptTen
- Two Machine Training
- Training on AWS
- Model and Dataset Stats
- Future Scope

# Problem Description

- Interpolate pollution values to unknown locations.
- GCN does this only for unknown neighbours.



Unknown Known Prediction

# Secret Sharing Protocols in CrypTen

---

- Private Addition: parties add their shares independent of each other.
- Private Multiplication: Beavers triples via trusted third party.
- Linear Functions: Done using private addition/multiplication.
- Non Linear Functions: Done using standard approximations and private addition/multiplication.

# Challenges : Porting to Newer Version of CrypTen

---


- Loss and gradients were abruptly going to very large values after 40 epochs.
- After investigation, we observed following:
  - Bug on multiplying zero tensors for parties greater than 2.
  - Exploding loss and gradient occurred in multi party setting.
  - Lowering learning rate lead to delay in explosion.
  - CrypTen library is under development phase and still evolving.
  - Code written for older CrypTen made compatible to newer version of CrypTen.

# Zero Tensor Multiplication Bug

```
In [3]: import crypten.mpc as mpc
import crypten.communicator as comm
from crypten.communicator.communicator import _logging

zero_array = np.zeros((5,5))

@mpc.run_multiprocess(world_size=5)
def check_zero():
    rank = comm.get().get_rank()
    zero1 = crypten.cryptensor(zero_array[0],src=0)
    zero2 = crypten.cryptensor(zero_array[0],src=0)
    zero3 = zero1*zero2
    crypten.print(zero3.get_plain_text())
check_zero()
```

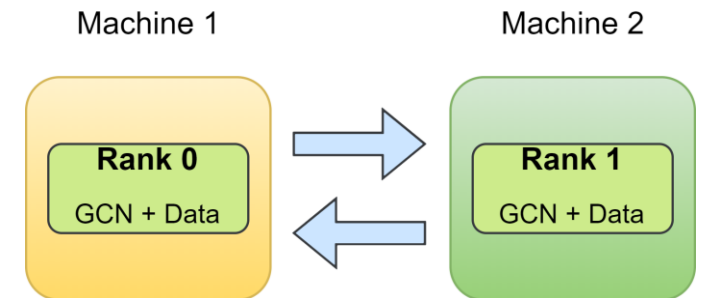


```
tensor([-4.2950e+09,  1.5259e-05,  0.0000e+00,  1.5259e-05, -4.2950e+09])
```

```
Out[3]: [None, None, None, None, None]
```

# Description : Two Machine Training

- CrypTen model was earlier trainable on one machine.
- To train in 2 machines, specified environment variables defining
  - World Size
  - Rank
  - IP address & port of Master Node
  - Communication Backend (GLOO/MPI/NCCL)
- Machine Configuration:
  - Machine 1: Intel Core i7 10750H, 16GB RAM
  - Machine 2: Intel Core i5 6200U, 8GB RAMAverage Ping Latency between two machines was 7.9 ms.



# Measurement : Two Machine Training

- Computation and communication cost measurement during training.
- Overhead on both machines was the same.

Rank	Packets	Data Transfer (MB)	Time to train per Epoch (s)	Time Packets Captured	Bandwidth (MB/s)	CPU Overhead (%)	Memory Overhead (%)
0	1002480	1685.89	4.14	844.239	1.997	39.61	1.98
1	1015286	1683.56	4.13	835.239	2.016	41.19	1.90

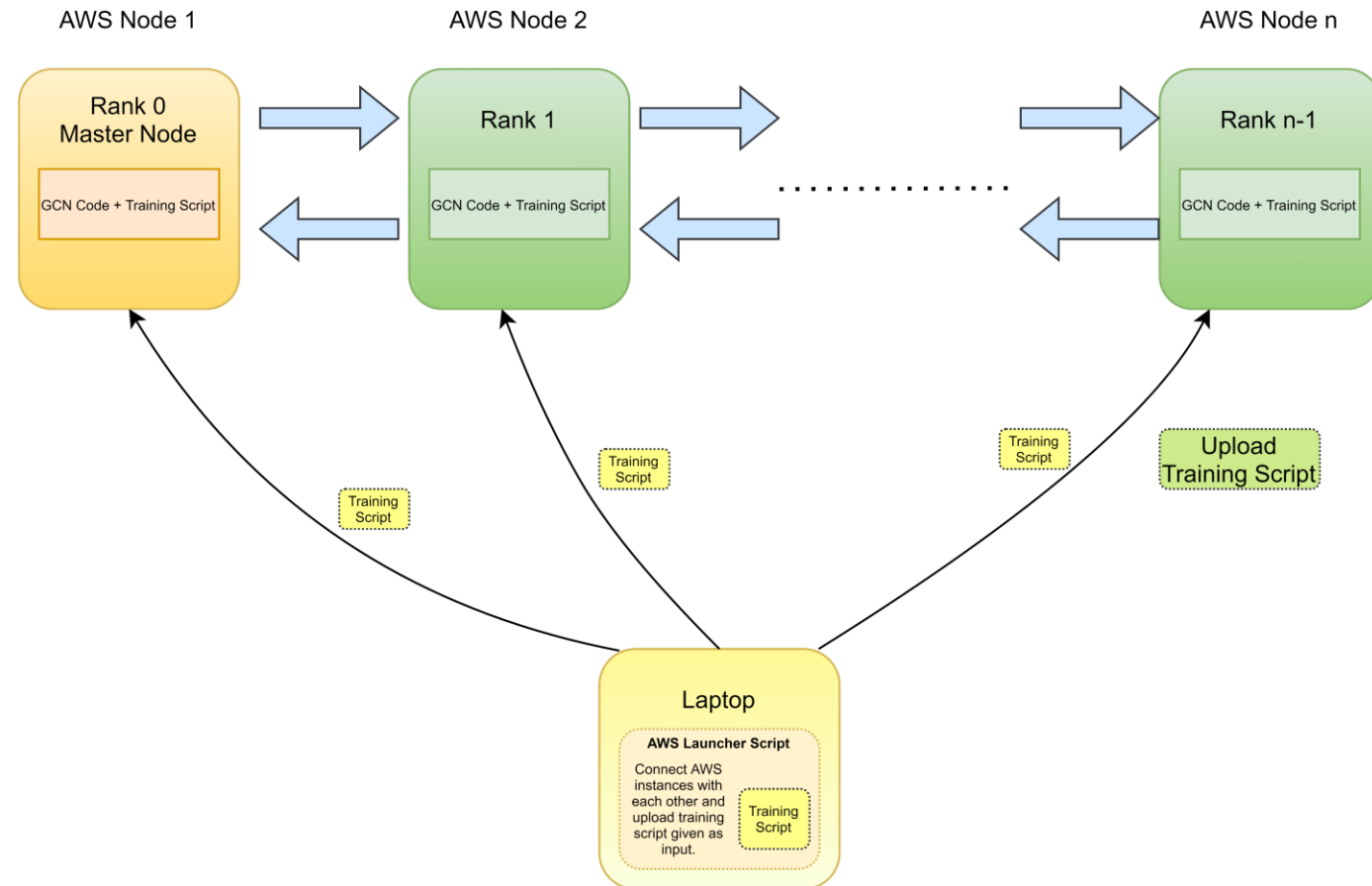


# Description : Training on AWS Nodes

---

- Not scalable to manually define environment variables on  $n$  AWS Nodes.
- Used AWS Launcher Script provided by CrypTen to handle this issue.
- AWS Instances used were *Deep Learning AMI (Ubuntu 18.04) Version 52.0*.
- Took the similar measurements for AWS and found equal overhead on each node.

# Training on AWS Nodes



# Training on AWS Nodes: Measurements

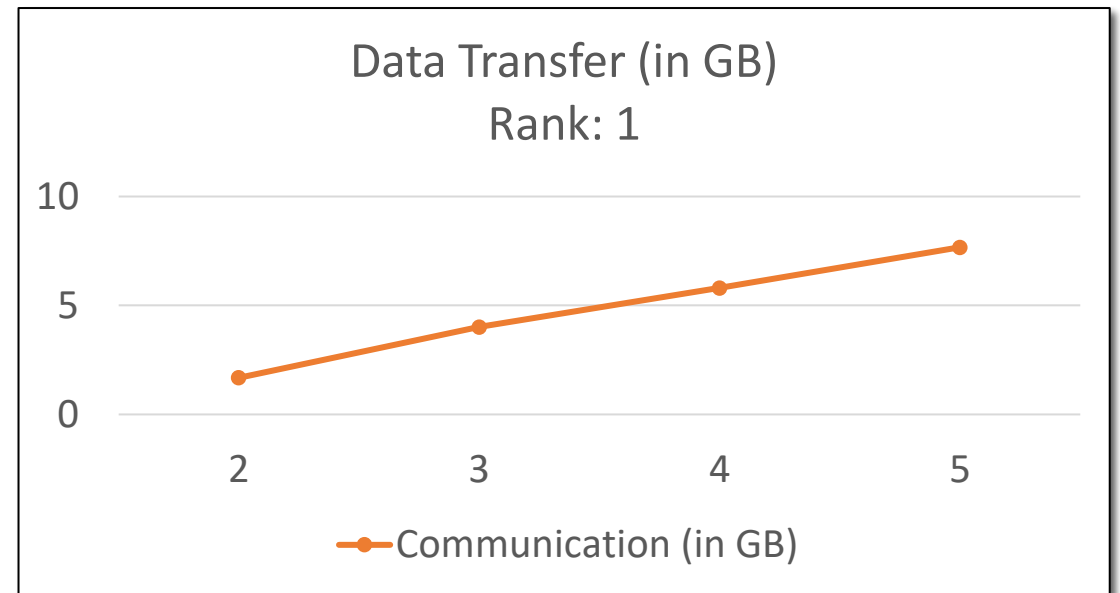
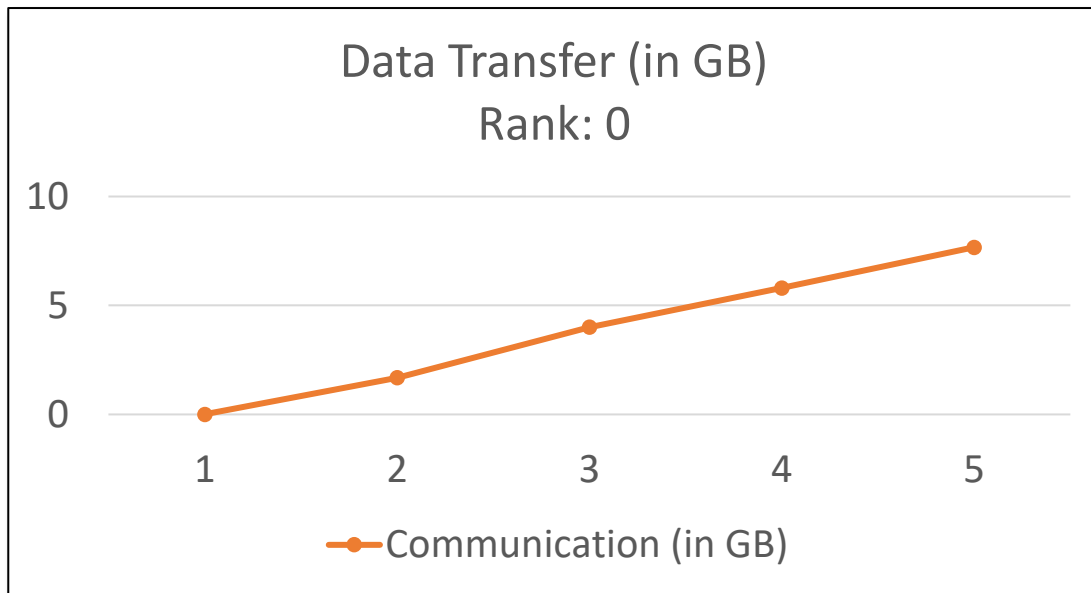
Measurements on rank 0 or master node

Wall Clock Time (s)	Training Time/Epoch (s)	Communication (in GB)	Communication (in MB) (per epoch)	CPU Overhead (%)	Memory (%)	Packets Captured
108.290	0.448	0.000	0.000	98.696	29.937	0
176.000	0.776	1.681	8.404	80.757	29.083	939229
427.930	1.983	4.002	20.008	53.739	29.839	4054223
549.300	2.541	5.803	29.014	48.521	29.445	6383688
923.360	4.382	7.669	38.343	41.745	29.553	10856337

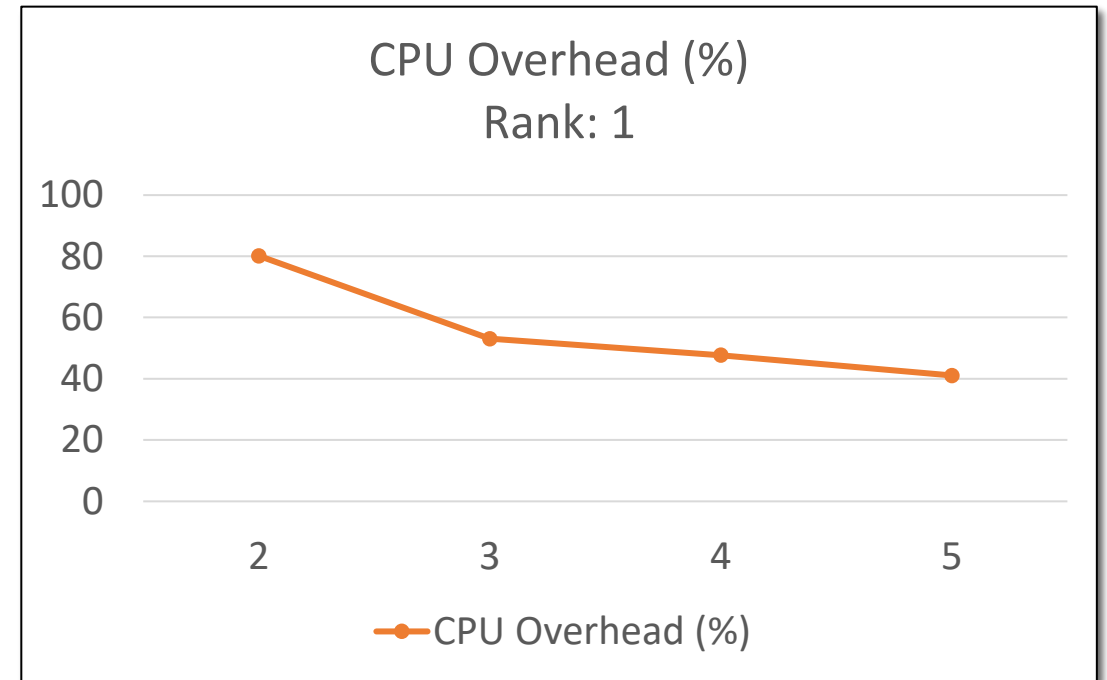
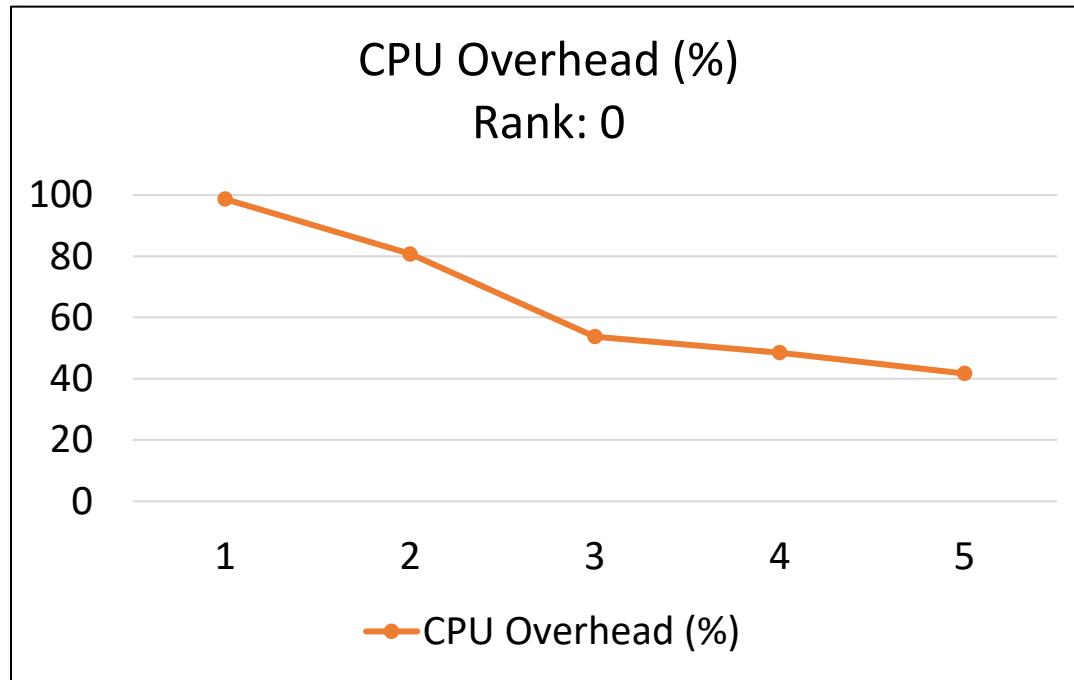
Measurements on rank 1 node

Wall Clock Time (s)	Training Time/Epoch (s)	Communication (in GB)	Communication (in MB) (per epoch)	CPU Overhead (%)	Memory (%)	Packets Captured
179.360	0.777	1.686	8.430	80.112	29.478	935784
427.920	1.987	4.008	20.039	53.066	29.158	4066562
555.200	2.581	5.802	29.011	47.692	29.892	6364673
956.170	4.537	7.670	38.352	41.066	29.318	10881596

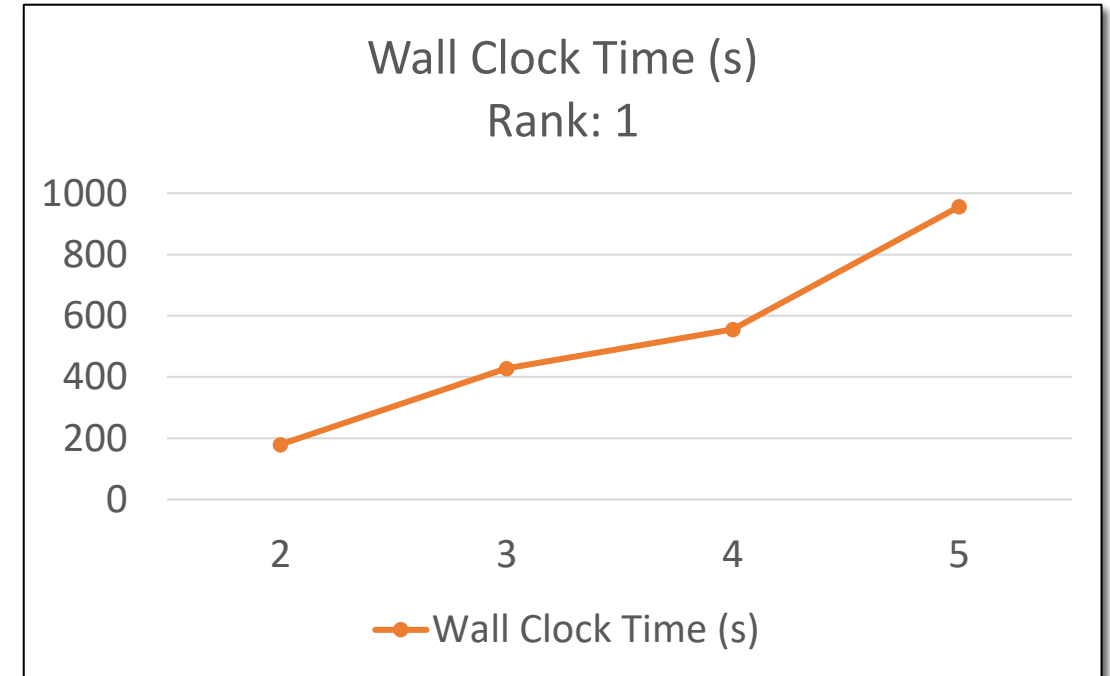
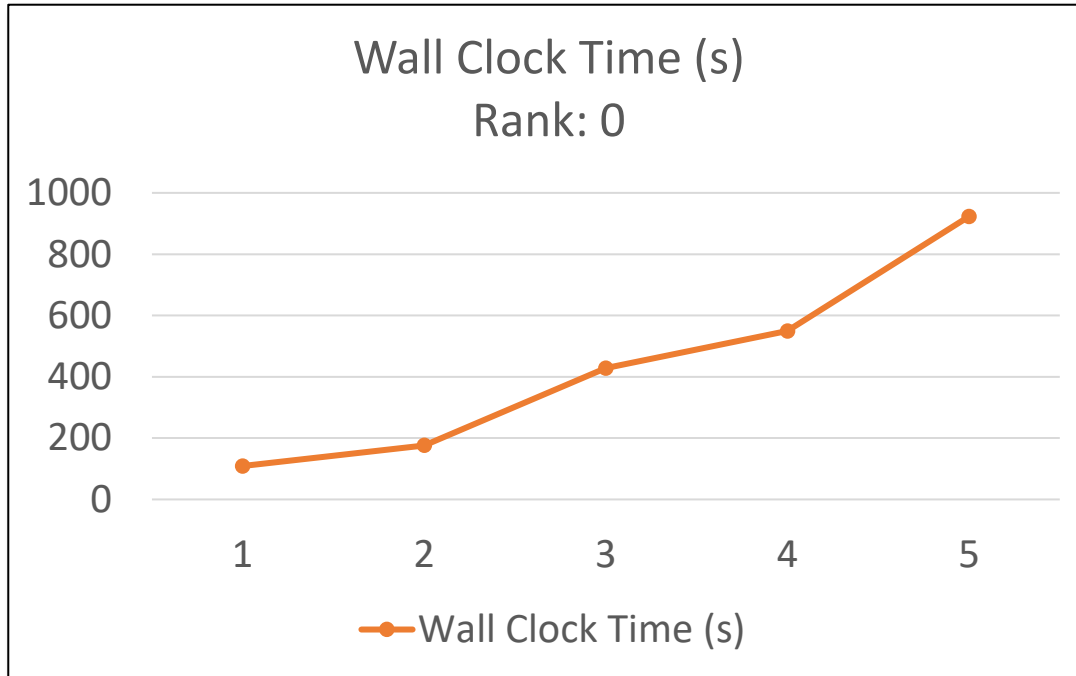
# Comparison b/w Rank 0 & 1: Data Transfer (in GB)



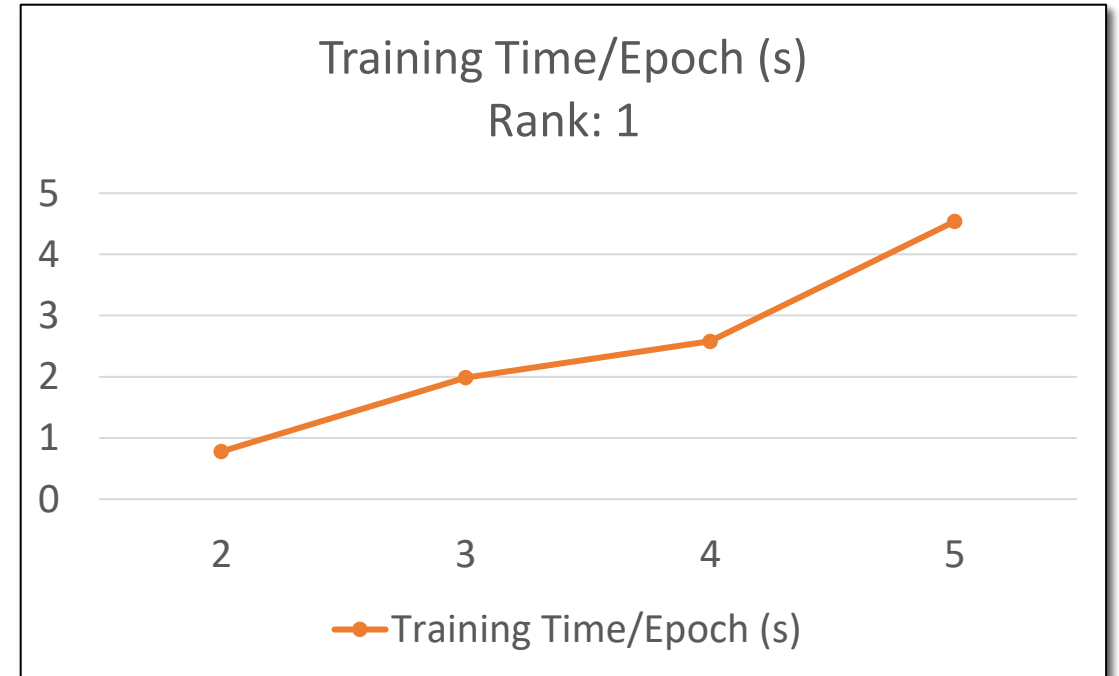
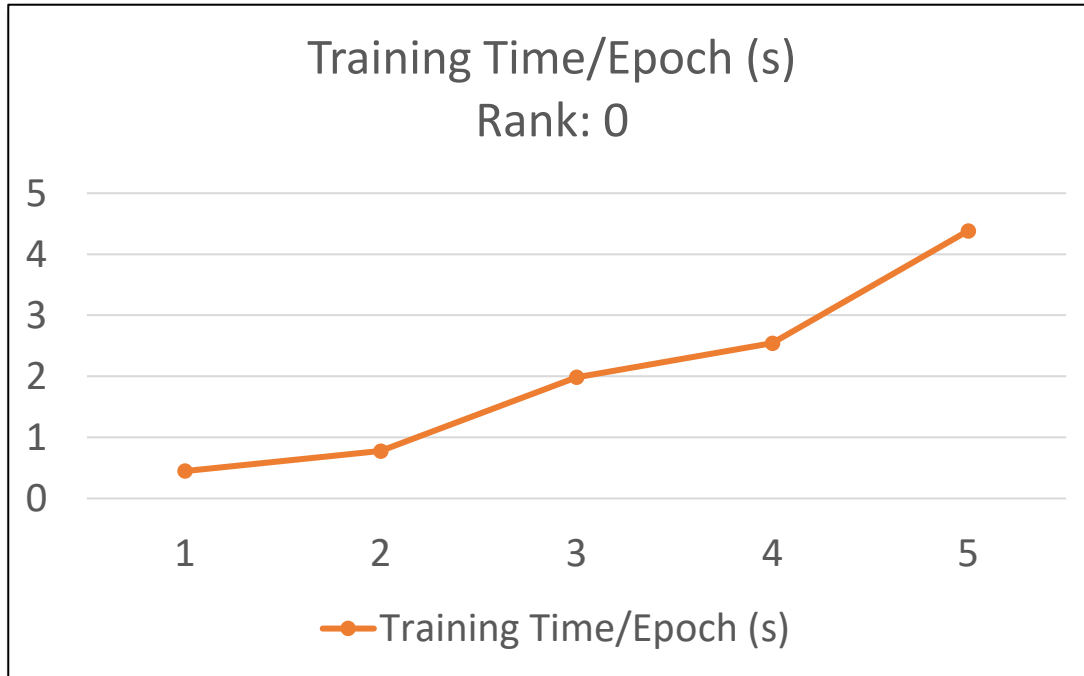
# Comparison b/w Rank 0 & 1: CPU Overhead



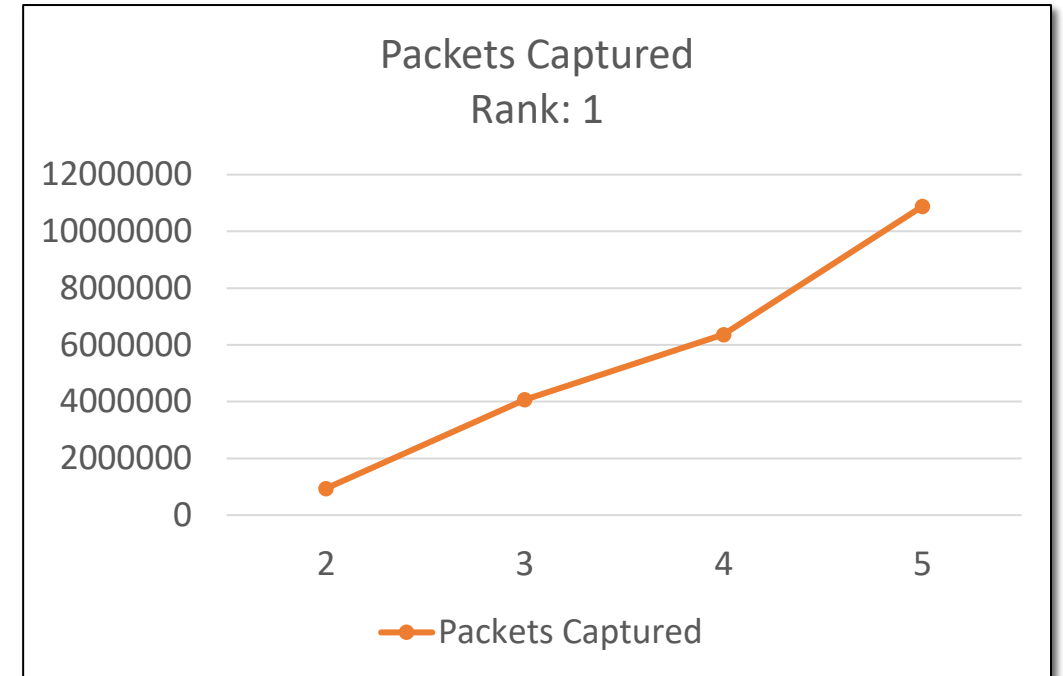
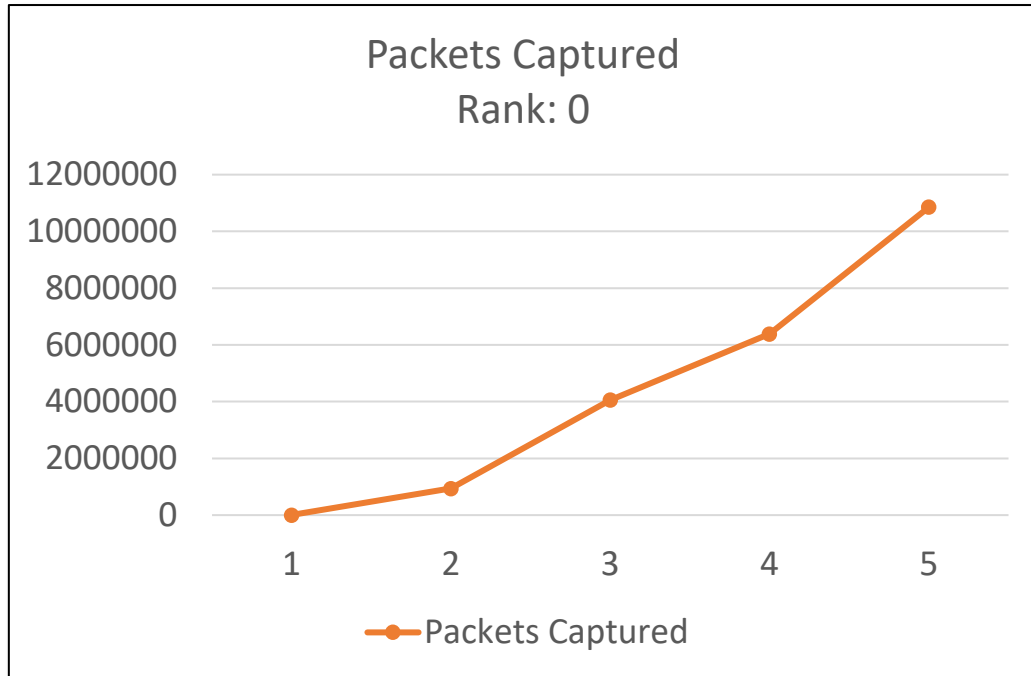
# Comparison b/w Rank 0 & 1: Wall Clock Time (s)



# Comparison b/w Rank 0 & 1: Training Time/Epoch (s)



# Comparison b/w Rank 0 & 1: Packets Captured





# Inference : Training on AWS Nodes

---

- CPU Overhead decreases with more parties.
- Wall clock time, training time per epoch and packets follow exponential trend.
- Memory overhead (~320 MB) remains constant with number of parties.
- Communication cost is linear in number of parties.

# Size : Model and Dataset Stats

---

Size of models and dataset

Model/Data	Size
1 Hour Dataset	0.14 MB
PyTorch GCN	2.3 kB
CrypTen GCN	2.6 kB

# Model and Dataset Stats: Accuracy

Train and Test RMSE Values

Model	Train_RMSE <sub>100</sub>	Test_RMSE <sub>100</sub>	Train_Time <sub>100</sub>	Train_RMSE <sub>FULL</sub>	Test_RMSE <sub>FULL</sub>	Train_Time <sub>FULL</sub>	VAR
GPR	29.76	30.41	1176.45	29.13	29.74	3433.04	Yes
Variational GPR	32.42	33.05	267.89	29.89	30.63	1053.88	Yes
ANN	47.04	47.51	51.78	31.61	32.49	141.42	No
GraphSAGE	-	35.45	4654.64	-	34.15	4891.81	No
Meaner	-	-	-	-	33.84	1578.64	No
GCN	109.95	49.60	0.438	40.211	50.08	4.22	No

**Note:** Full Training means training the model until convergence, hence number of epochs in full training may be different for each model.

# GP for Interpolation

---

We also tried porting GP based on GPyTorch to CrypTen but there were few roadblocks:

- Uses LazyTensor, not supported in CrypTen.
- Covariance Matrix calculations not supported.
- Bool Operations incompatible with MPC Tensor.
- Variational ELBO loss not supported.
- CrypTen does not support in-place operations and features like `retain_graph=True` in Autograd.

# Future Scope

---

## **Future Models:**

- GCN model has higher RMSE but low training time.
- Low training time is attributed to very low size of model and dataset.
- Develop larger and more sophisticated GCN models which may result in low RMSE values and training time.

## **CrypTen:**

- Communication and memory overhead is huge even for model and data size in kBs, which may increase further for more complex models.

---

Thank You!