DISTRIBUTED ML Training: A SERVERLESS ARCHITECTURAL APPROACH

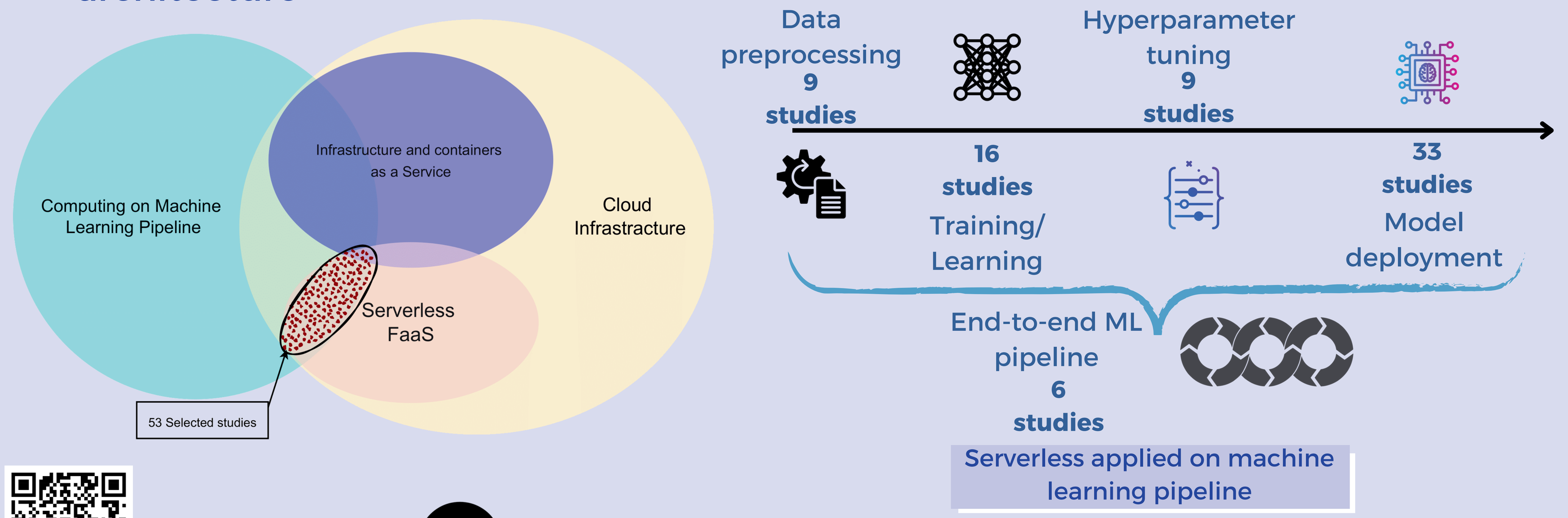
Amine Barrak
Fehmi Jaafar (Advisor)
Fabio Petrillo (Co-Advisor)

Serverless Functions & Distributed ML: Challenges, Opportunities, Best Practices

- How have serverless functions been utilized in ML pipelines ?
- How can serverless functions be used to speed up ML training?
- How can communication overhead be mitigated in serverless distributed training?
- How can we secure and ensure fault tolerance in serverless training?
- How do we propose a fully serverless ML training architecture?
- What did we learn from comparing serverless ML frameworks?

Serverless Functionsutilization in Machine Learning Pipelines

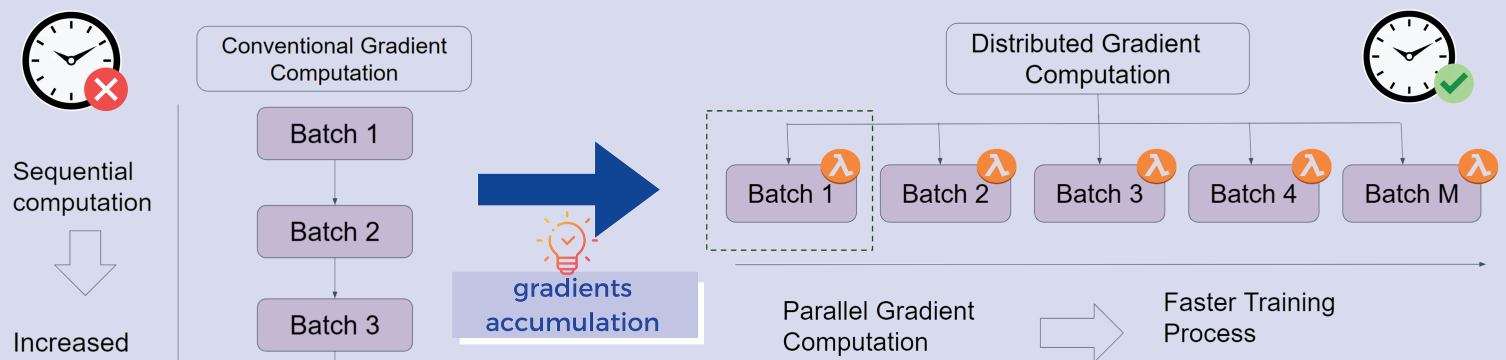
Conducting a systematic mapping study on ML systems applied on serverless architecture



IEEE Access '22

AmineBarrak/Serverless-on-ML

Accelerating ML training with serverless functions



Time improvement by 97.34% but cost 5.3 times more than traditional server based methods.

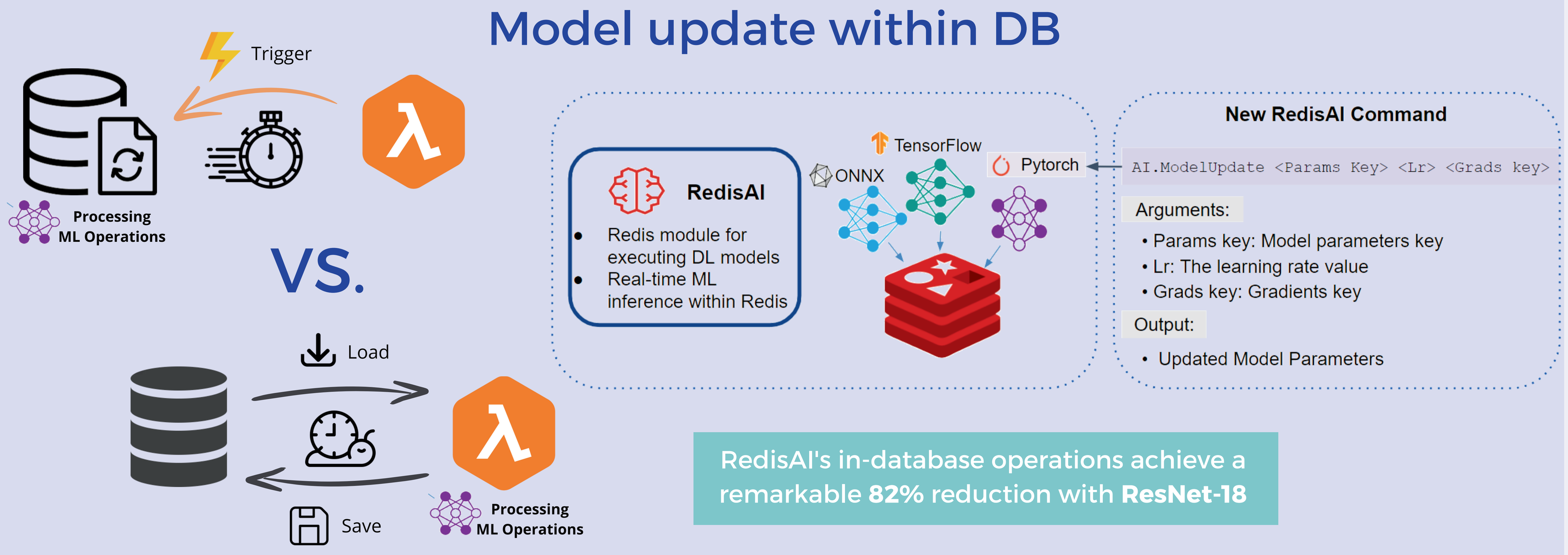
IC2E' 23

AmineBarrak/PeerToPeerServerless

Reduce communication overhead: Perform ML operations within the database

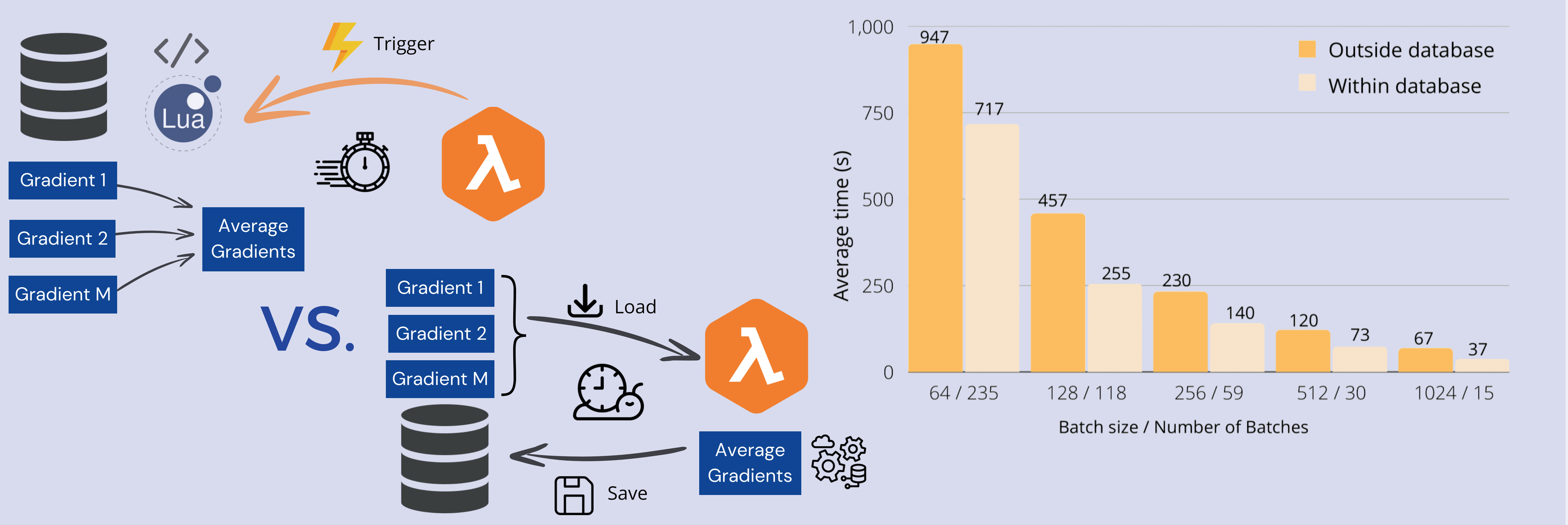
AAAI' 24

Model update within DB



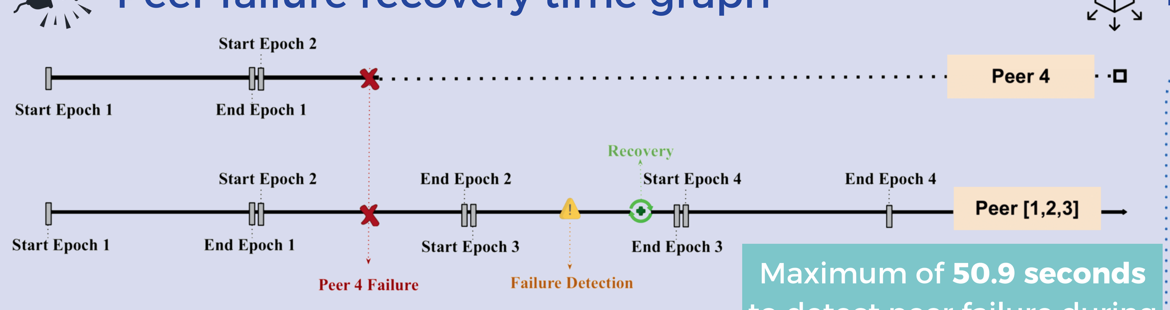
RedisAI's in-database operations achieve a remarkable 82% reduction with ResNet-18

Gradients Averaging within DB



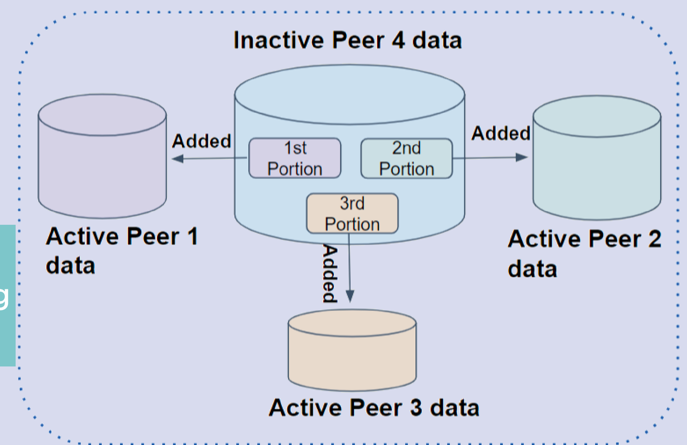
Secure & Fault tolerant communication in serverless distributed training

Peer failure recovery time graph



Maximum of 50.9 seconds to detect peer failure during the heartbeat step

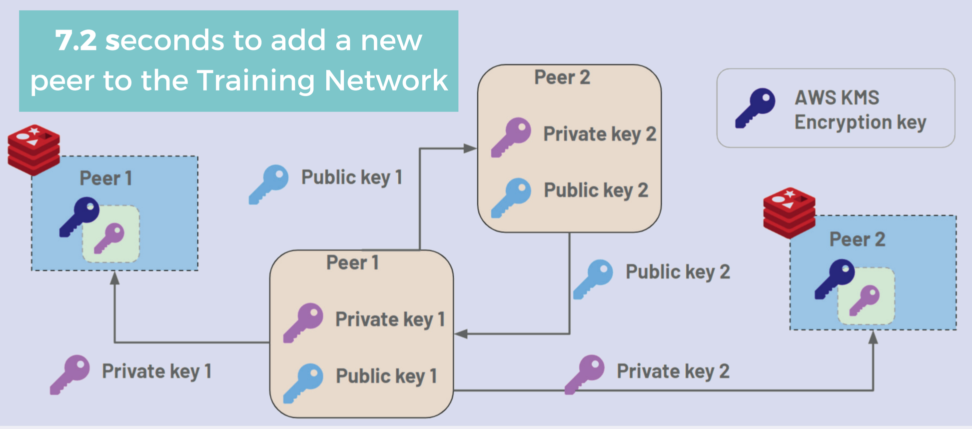
Data redistribution of failed peer



Adding new peers

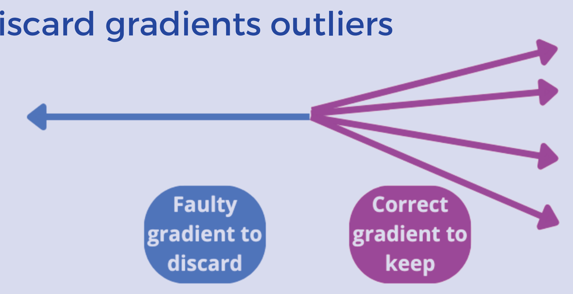
Peers authentication mechanism / Asymmetric Encryption

7.2 seconds to add a new peer to the Training Network



Tolerating ML Byzantine attacks

Discard gradients outliers



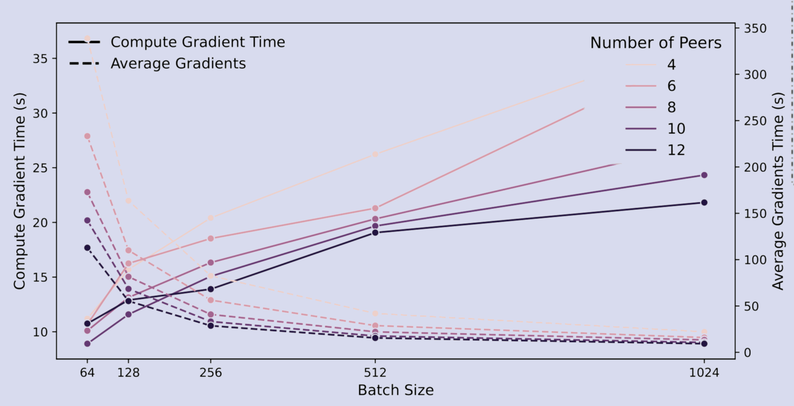
SPIRT : Framework for training ML workflow in serverless environments

- Intra-peer Scalability: parallel gradients computation
- Inter-peer Scalability: Changing number of peers

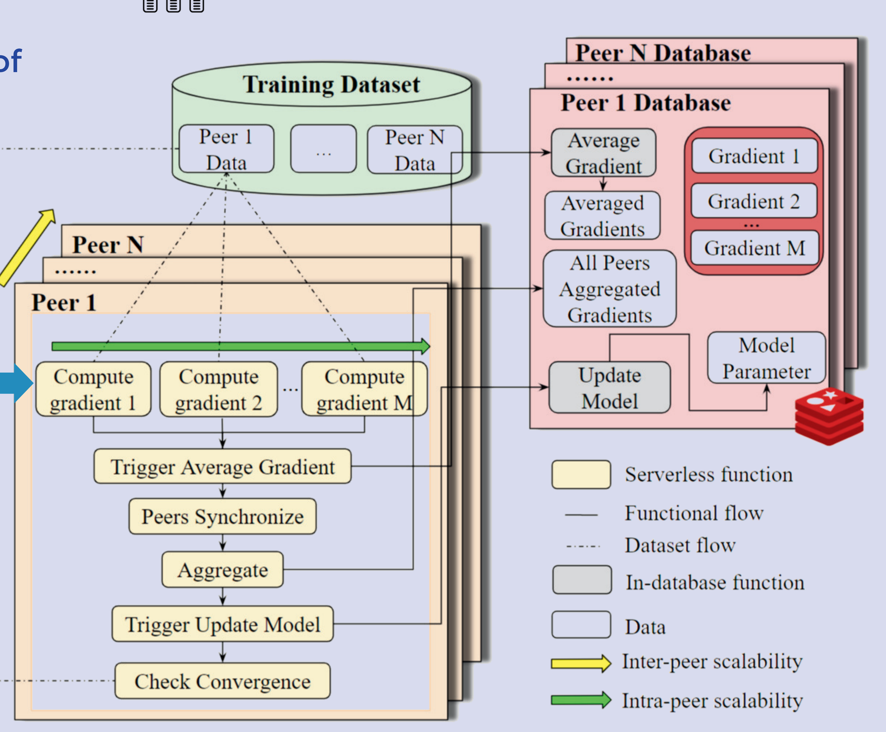
AWS step function for epoch orchestration

No single point of failure: robust peer-to-peer (P2P) architecture

Varying Batch Sizes for Different Number of Peers with DenseNet-121 Model



SPIRT Framework



QRS' 23

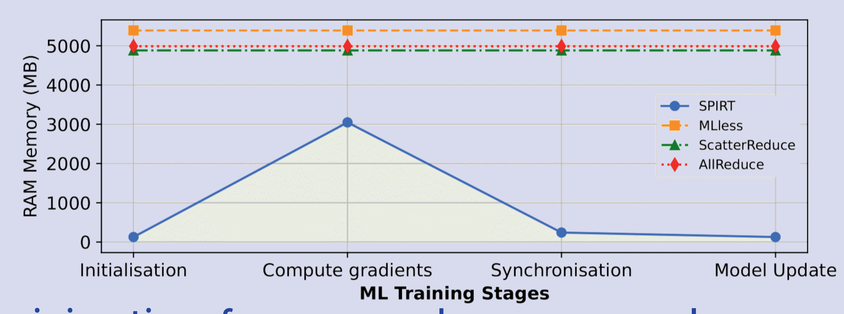
AmineBarrak/TrainingML_P2P_Serverless

Serverless ML training frameworks comparison

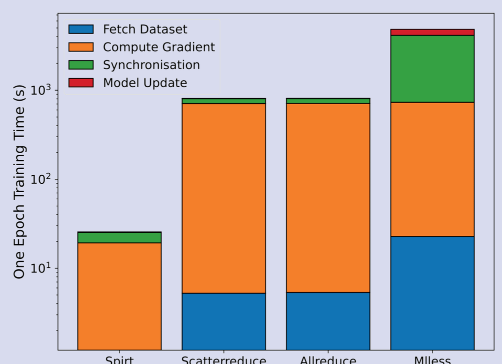
Typical Serverless Training Workflow

- **Fetch dataset** : Each worker fetches its dataset partition.
- **Parallel Compute gradients** : Independent training of workers on local batches.
- **Synchronization** : Gradients upload and aggregation in shared database by workers.
- **Model Update** : Update each local model with the aggregated gradients.

Split the training workflow into separate serverless functions: Optimisation in RAM usage



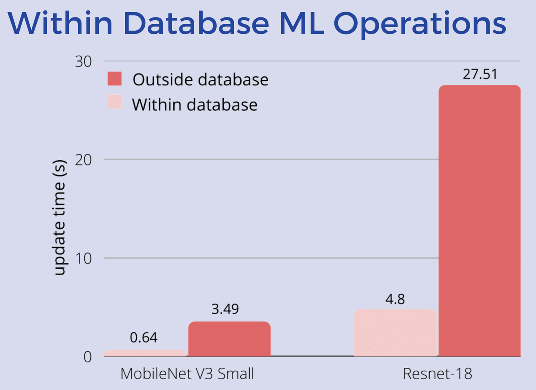
Training time for one epoch across serverless frameworks



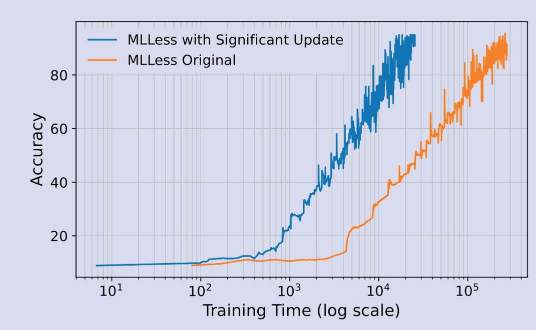
Preprint TPDS 2024

Reduce Communication Overhead

- SPIRT: Within Database ML Operations



- MLLESS: Synchronise only Significant Updates



- LambdaML: ScatterReduce Vs. AllReduce

UQAC UNIVERSITÉ DU QUÉBEC

Electronic Poster Version.

IPDPS 2024 • San Francisco, California USA
San Francisco, California • May 27-31, 2024