inovex

Parallelized Training of Deep NN

Comparison of Current Concepts and Frameworks

Sebastian Jäger, Hans-Peter Zorn,
Stefan Igel, Christian Zirpins

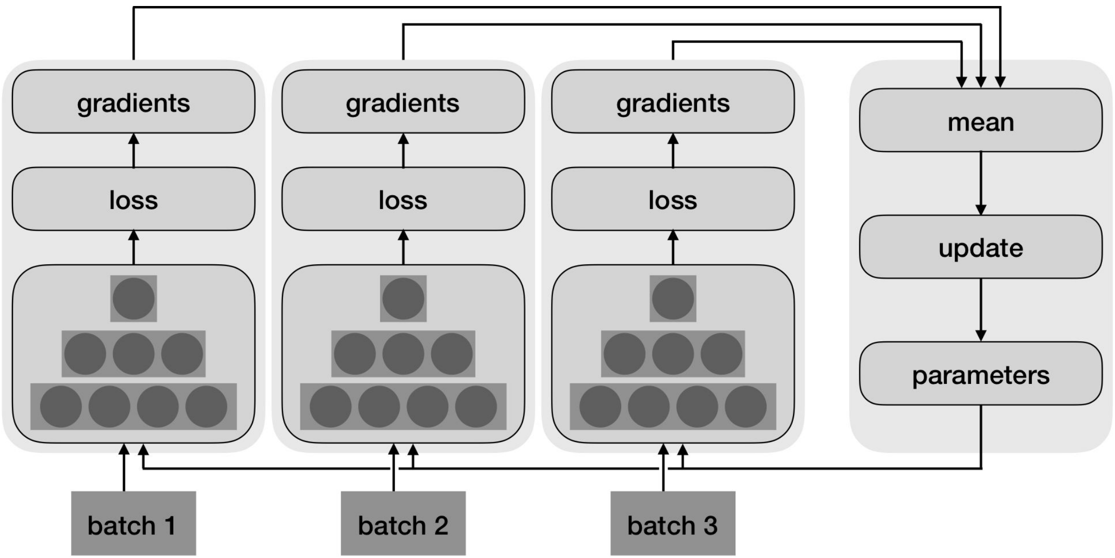
Rennes, Dec 10, 2018

Motivation

- Need to scale the training of neural networks horizontally
- Kubernetes based technology stack
- Scalability of concepts and frameworks

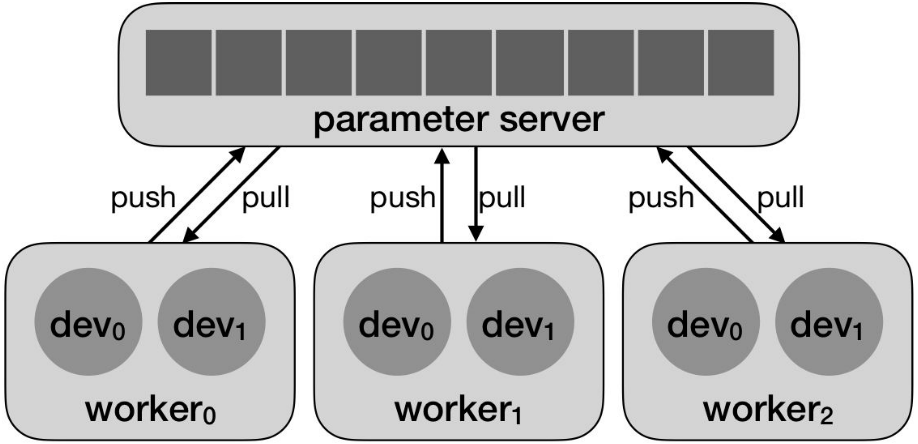
Distributed Training Methods

Data Parallelism



Data Parallelism

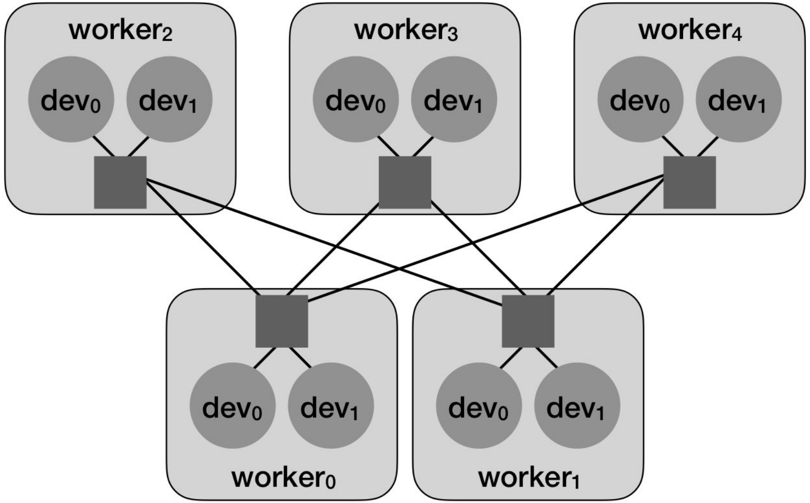
Centralized Parameter Server



TensorFlow: https://www.tensorflow.org

Data Parallelism

Decentralized Parameter Server



mxnet

Experimental Setup

Environment

- Google Kubernetes Engine
- CPU: 2.6 GHz

- Ubuntu 16.04
- TensorFlow 1.8.0
- MXNet 1.3.0

Experimental Setup

Networks

Convolutional NN

- LeNet-5
  - 5 layer
  - 10 classes
- Fashion MNIST
  - 28x28 gray-scale

Recurrent NN

- LSTM
  - 2 layer
  - 200 units
- Penn Tree Bank
  - 1.000.000 words

Experimental Setup

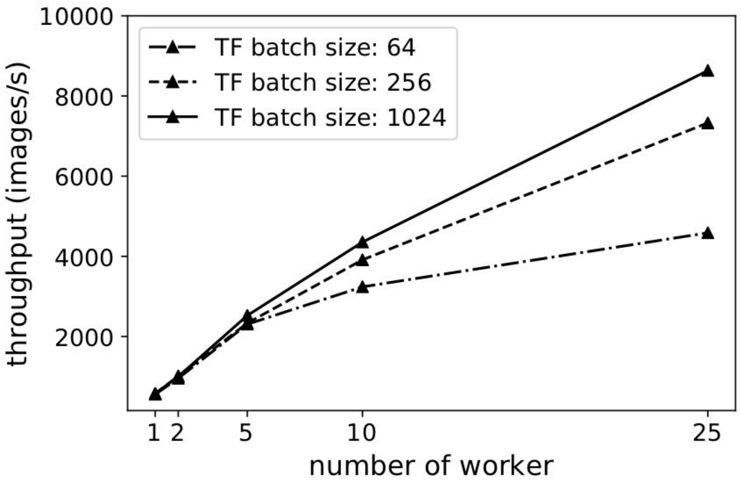
Metrics

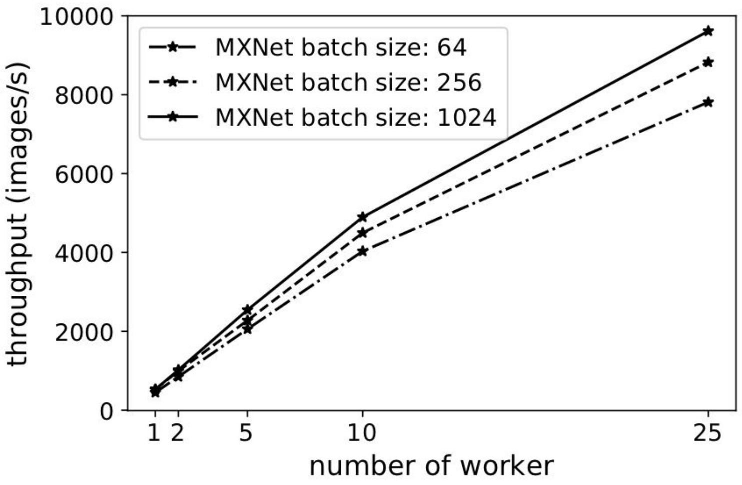
$$throughput_n = \frac{no.\ examples * epochs * no.\ workers}{training\ time_n}$$

$$speedup_n = \frac{throughput_n}{throughput_1}$$

Results

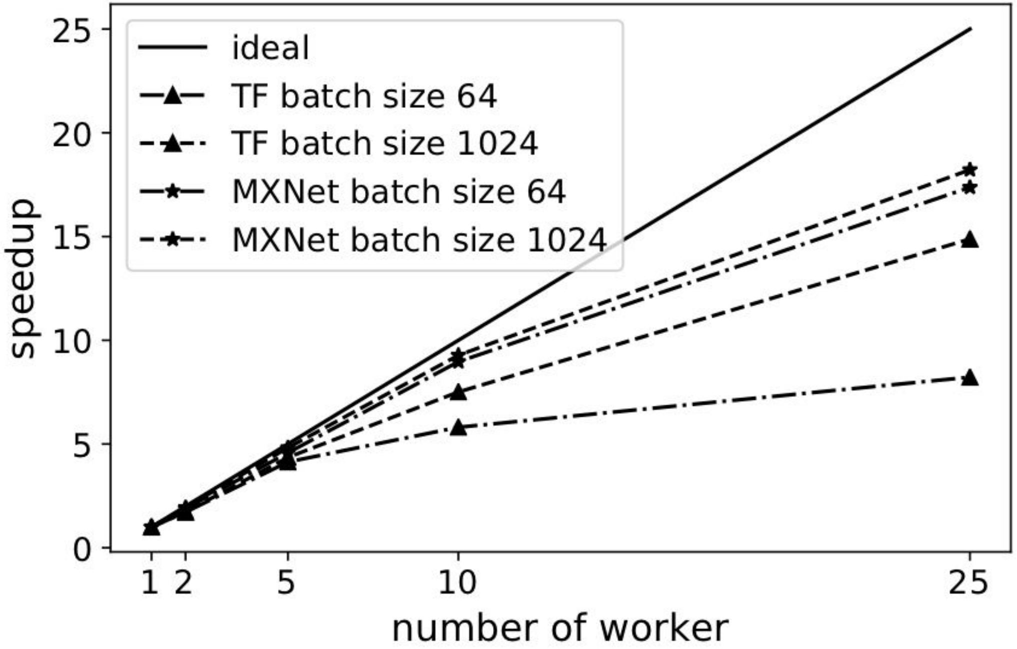
Convolutional Neural Network





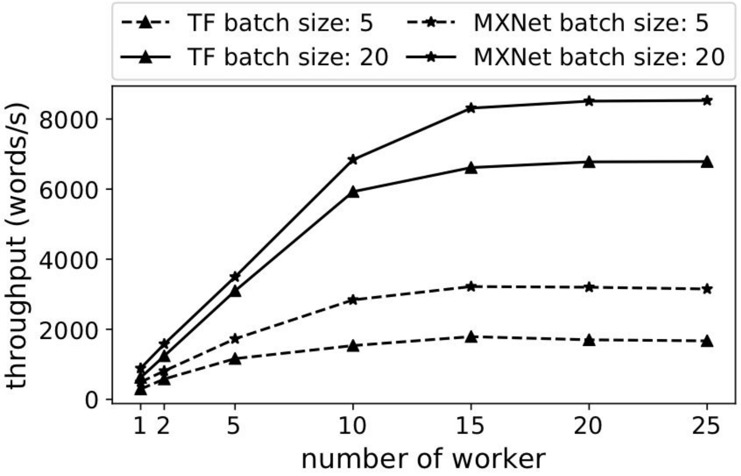
Results

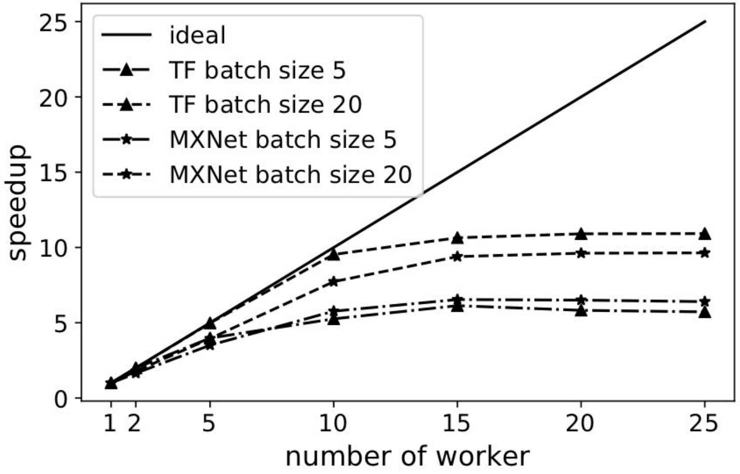
Convolutional Neural Network



Results

Recurrent Neural Network





Summarizating the Experiments

Decentralized Parameter Server ...

- more robust regarding increasing communication effort
- scales better for small NN

For bigger/ more complex NN ...

- no significant difference between concepts

Conclusion

MXNet ...

- for small NN better scalability and throughput
- for bigger NN higher throughput
- less and less complicated code
- easier to scale up training

Thank you

Sebastian Jäger
@se_jaeger

inovex GmbH
Ludwig-Erhard-Allee 6
76131 Karlsruhe

sebastian.jaeger@inovex.de