Deep Learning Platform Comparisons & A Predictive Model for Real-Time Offline Application.

By Timileyin Okoya

Objective

1. Compare the various available platforms for deep learning and subsequently predict the Torque supplied by an exoskeleton, based on the available features in an offline setting at real time.

1. Predict the Reference Torque based on the features derived from the IMU sensory system and in the exoskeleton and save it with a protocol buffer that can be used for real-time offline prediction.

Introduction

What are Deep Learning Platforms?

Definition: Software frameworks designed to facilitate the development, training, and deployment of deep learning models.

Components: Typically include tools for data preprocessing, model building, training, evaluation, and deployment.

Popular Deep Learning Platforms: Tensorflow, PyTorch, Keras, MXNet etc.

Introduction

A group of researchers performed a comparative evaluation of three prominent deep learning frameworks: TensorFlow, Caffe, and Torch. Their analysis included metrics such as accuracy, runtime performance, and the frameworks' adaptability to various datasets.

A team of researchers conducted a benchmarking analysis of four advanced deep learning platforms—Caffe, CNTK, TensorFlow, and Torch—across three categories of neural networks.

Researchers carried out a comparative examination of deep learning frameworks, encompassing Caffe, neon, Theano, and Torch. Their investigation centered on three principal facets: 1. Extensibility, 2. Hardware utilization, and 3. Speed, which encompassed gradient computation time (training time) and forward time (testing time).

NB: The comparison in this exoskeleton use case is tailored towards the requirements

Legacy frameworks

- CNTK
- Tensorflow
- Torch
- Caffe

P.S.: The frameworks listed above were investigated to determine their suitability for the tasks proposed. The choice of these frameworks was first and foremost streamlined to their proprietary license. There was a huge bias for Open-Source frameworks and subsequently the accessibility of their libraries via C++ and/or Python wrappers

Legacy Framework Comparison

	Opensource	Python	C++	GPU	Cross Platform
Cafe	Y	Y	Y	Y	Y
Torch	Y	Y	Y	Y	N
CNTK	Y	Y	Y	Y	N
Tensorflow	Y	Y	Y	Y	Y

Legacy Framework Comparison

	Protocol Buffer	Mobile	Embedded systems	Cloud	Restful APIs
Caffe	N[1]	Y	Y	Y	N[2]
Torch	N	N[3]	N[3]	Y	N[4]
CNTK	N[5]	N	N	Y[6]	N
Tensorflow	Y[14]	Y	Y	Y	Y

Legacy Framework Comparison

1 Caffe2 has some sketchy support for saving models in a protocol buffer, some versions of python are supported and latest python3 is barely supported. Using Caffe2 with Python 2.7 is an adventure at the moment.

2 Caffe2 doesn’t have an out-of-the-box deployment for Restful APIs. There are a couple of open source workarounds but nothing definitive. There is a good implementation from NVIDIA using docker and Go8.

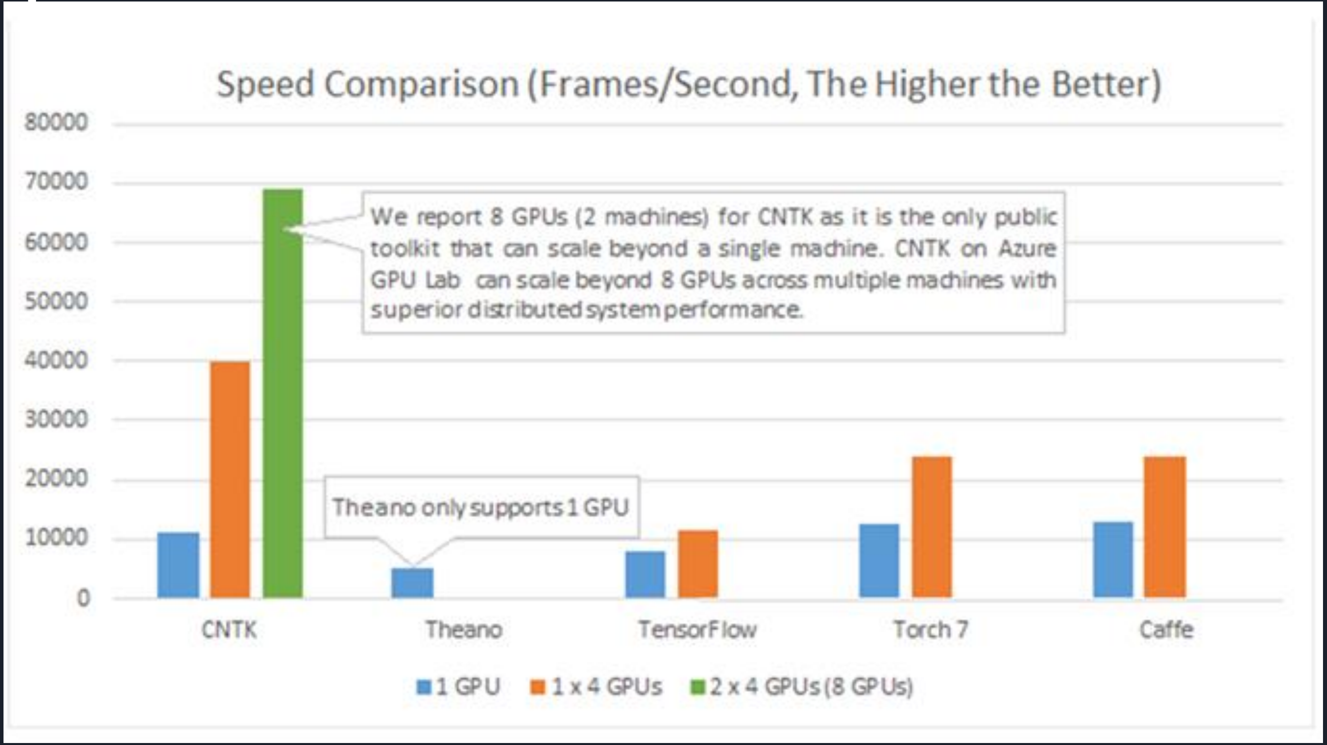
3 Torch doesn't have an out-of-the-box deployment for mobile. Although deployments on mobile are possible by transferring from Pytorch to Caffe2 using ONNX9. Similar workarounds are available for embedded systems depending on the platforms been investigated.

4 Torch has nothing definitive for Restful APIs. The only way around this is to build a custom web server.

5 CNTK does not currently save in protocol buffers but they have added this feature in their version 2 which is in it’s beta version at the moment.

6 CNTK has support for Azure and limited for other cloud platforms.

Comparison based on speed & performance



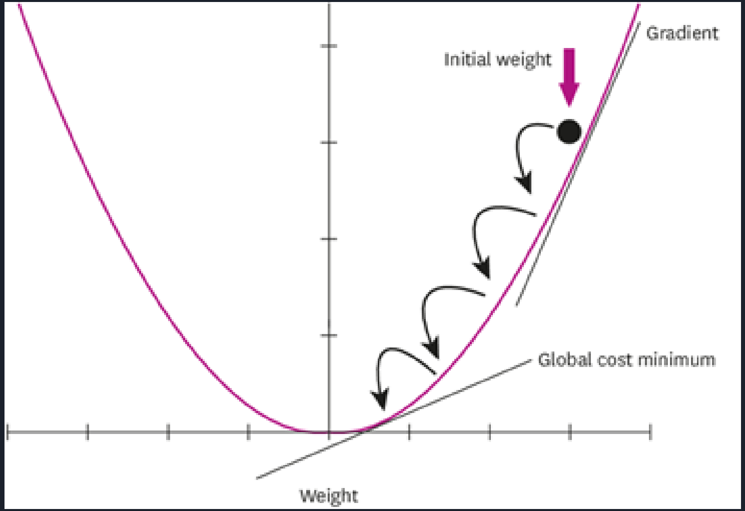
Tensorflow

- An end-to-end platform for machine learning
- TensorFlow makes it easy to create ML models that can run in any environment
- Intuitive APIs
- Code samples
- Curated curriculum for learning
- Community

TensorFlow

Cost Function Optimization

- Gradient Descent Optimization
- Adam Optimizer
- Ada Gradient Optimizer



ADAM: A Method for Stochastic Optimization

An algorithm for first-order gradient-based optimization of stochastic objective functions, based on adaptive estimates of lower-order moments.

- The method is straightforward to implement, is computationally efficient, has little memory requirements, is invariant to diagonal rescaling of the gradients, and is well suited for problems that are large in terms of data and/or parameters.
- The method is also appropriate for non-stationary objectives and problems with very noisy and/or sparse gradients.

Neural Network

- Input dimension: 10
- Learning rate: 0.001
- Adam Optimiser: default

Dataset

- The first three columns(time, Mcmd and McmdRaw) were dropped.
- NaN and Inf were replaced with 0 and -1 respectively.
- Training: 3000 rows
- Testing: 231 rows

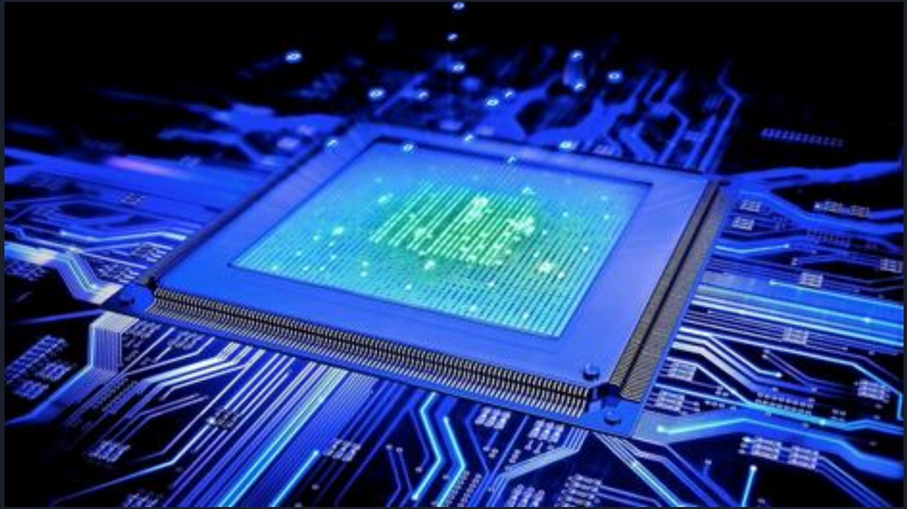
Computing specifications

Intel® Core™ i7-7700K CPU @ 4.20GHz × 8

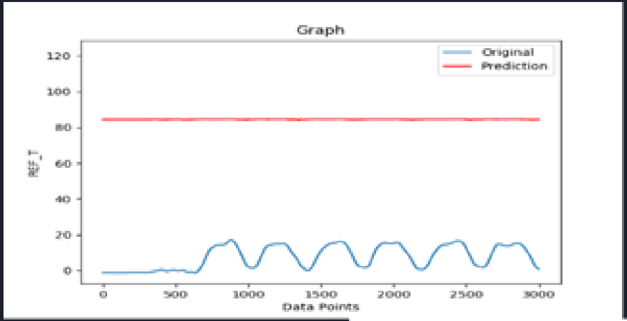
NVIDIA Quadro P2000/PCIe/SSE2,

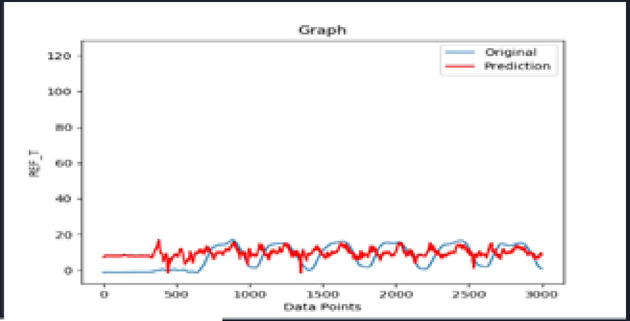
64 bit architecture

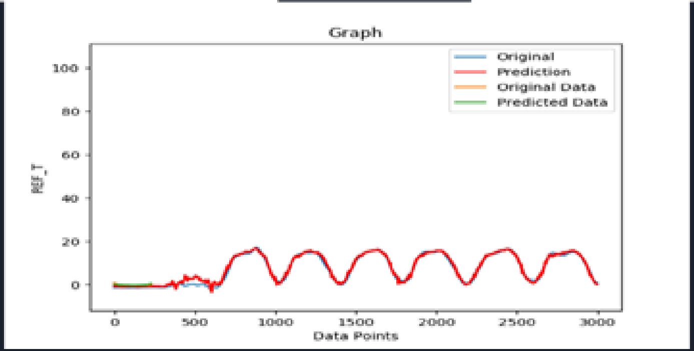
RAM: 15.6 GB



Results: Convergence of the learning process during training







Results: Torque from saved model

```
Original : [0.98017592]  Predicted : [0.8431047]

Original : [0.96051325]  Predicted : [0.8122009]

Original : [0.94659665]  Predicted : [0.7813101]

Original : [0.91658556]  Predicted : [0.74948907]

Original : [0.89099949]  Predicted : [0.72170436]

Original : [0.8648422]  Predicted : [0.77050674]

Original : [0.83622914]  Predicted : [0.72312975]
```

Next steps

- Use saved model in an android app
- Use saved model interfacing with a HTTP web server via REST APIs.

Conclusion

Platform Comparison:

- Conducted an in-depth comparison of various deep learning platforms.
- Evaluated based on specific criteria tailored to the project's unique requirements.

Predictive Model:

- Developed a predictive model to estimate the Reference Torque using features derived from the IMU sensory system in the exoskeleton.
- Implemented a protocol buffer to store predictions for real-time offline use.

Questions & Answers

Thank you!

okoya.timi@hotmail.com