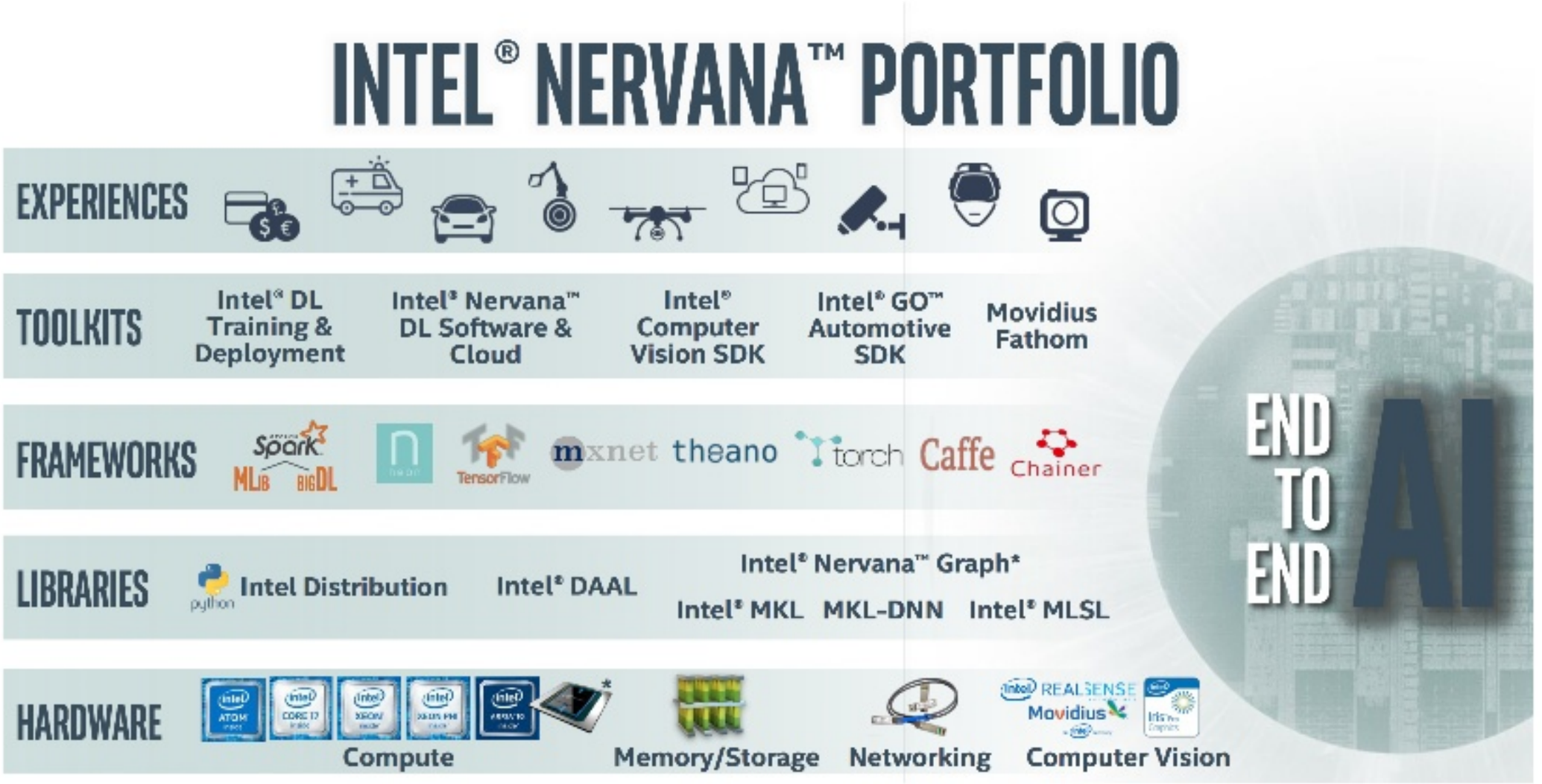
A Predictive Analytics Workflow on DICOM Images using Apache Spark

Anahita Bhiwandiwalla – Intel Corporation

Karthik Vadla – Intel Corporation

Who we are?


INTEL® NERVANA™ PORTFOLIO
EXPERIENCES
TOOLKITS
Intel® DL Training & Deployment
Intel® Nervana™ DL Software & Cloud
Intel® Computer Vision SDK
Intel® GO™ Automotive SDK
Movidius Fathom
FRAMEWORKS
Spark MLlib BigDL
TensorFlow
mxnet
theano
torch
Caffe
Chainer
LIBRARIES
python
Intel Distribution
Intel® DAAL
Intel® Nervana™ Graph*
Intel® MKL
MKL-DNN
Intel® MLSL
HARDWARE
Compute
Memory/Storage
Networking
REALSENSE
Movidius
Computer Vision
END TO END AI

Agenda

- Use-case overview
- Challenges
- What is our data?
- Deriving Insights
- Machine Learning
- Tools
- Spark-tk
- Demo
- spark-tensorflow-connector
- Performance

Use-case overview

- Vast amount of medical image data available
- Meta-data accompanying the image
  - Patient information
  - Status of condition
  - Method of image capture
  - Time to events(if part of a study)

Create distributed technology needed to efficiently store, scan and analyze data in healthcare.

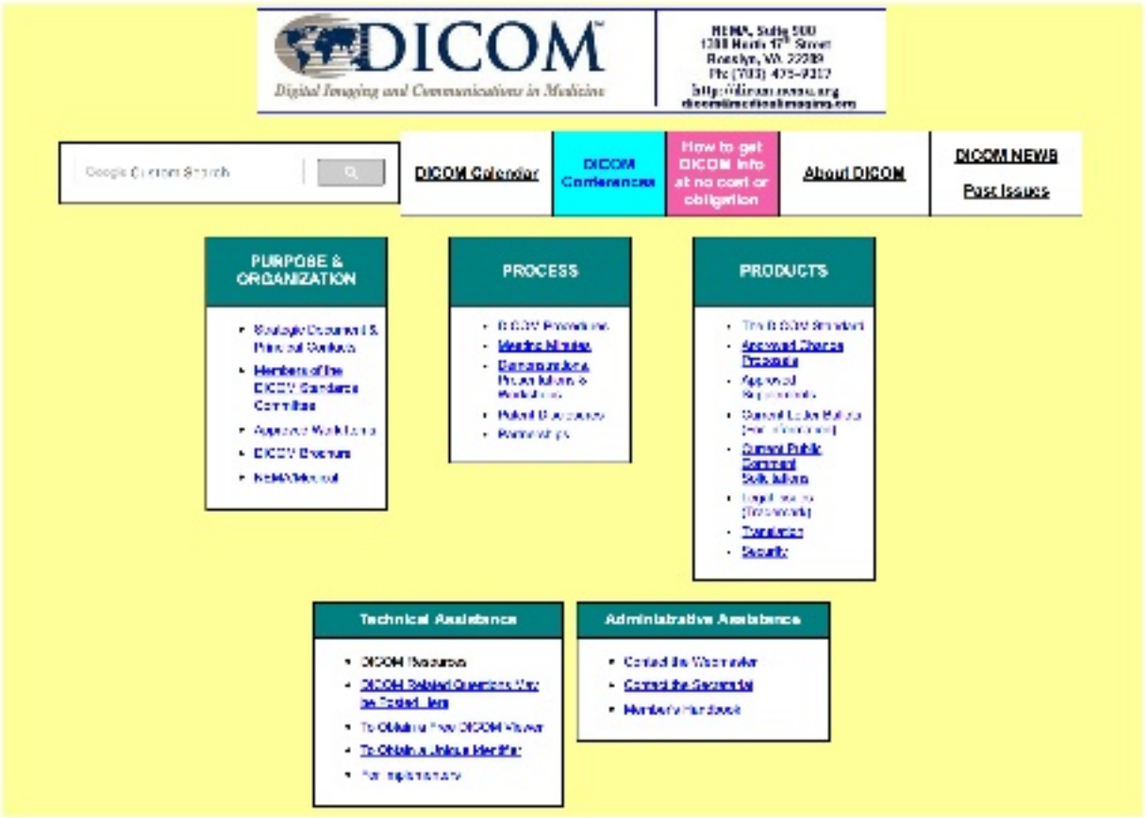
Challenges

- Getting data for experimentation is hard!
- Combine data from multiple public sources for simulation
- Multiple tools – image processing libraries, visualization tools, distributed engines, ML libraries
- Compatibility across different tools

How do we make Machine Learning easy to use?

Data: DICOM

- **D**igital **I**maging and **COM**munications in Medicine
- International standard format to store, exchange, and transmit medical images
- Standards for imaging modalities - radiography, ultrasonography, computed tomography (CT), magnetic resonance imaging (MRI), and radiation therapy.
- Protocols for image exchange, image compression, 3-D visualization, image presentation etc.



Data: Meta-data

- **Image** related: storage, date, relevant parts of the image etc.
- **Patient** related: age, weight, height, DOB, medical history
- **Device** related: manufacturer information, operator, method of capture

Deriving insights

- Can we develop smart filtering techniques based on image meta-data?
- Can we conclude and predict certain conditions based on patient meta-data?
- Can we classify image capture techniques and suggest improvements based on device meta-data?
- Can we combine image & patient meta-data to conclude and predict?
- Can analytics help automate/improve image capture techniques?

Machine Learning

- **Patient data**
  - Train classifier model on patient data to predict presence of a condition
    - Logistic Regression, Random Forest, Naïve Bayes
  - Train regressor model on patient data to predict probability of a condition
    - Linear Regression, Random Forest
  - Train survival analysis model on patient data to predict time to event, compare risk of populations
    - Accelerate Failure Time Model, Cox Proportional Hazards Model

Machine Learning

- **Image data**
  - Compute the Eigen decomposition of the image
  - Compute the Covariance Matrix of the image
  - Compute the Principal Component Analysis of the image pixel data

Machine Learning

- **Device data:**
  - Predict trends in image capture techniques
    - Classification/Regression techniques
  - Predict device time to failure
    - Cox Proportional Hazards Model, Accelerated Failure Time
  - Recommendations to device capture methods
    - Collaborative filtering

Tools

- **pydicom**
  - Python package for working with DICOM files such as medical images, reports, and radiotherapy
  - Single threaded
- **dcm4che**
  - Collection of open source applications and utilities for the healthcare enterprise.
  - Developed in the Java programming language for performance and portability, supporting deployment on JDK 1.6 and up
- **matplotlib**
  - Python 2D plotting library which produces publication quality figures in a variety of hardcopy formats and interactive environments across platforms.
  - Can be used in Python scripts, Python and IPython shell, jupyter notebook, web application servers
- **Machine Learning libraries**
  - Classifiers, regressors, failure analysis analysis
  - Eigen decomposition

Spark-tk Introduction

- Open source library enhancing the Spark experience
- APIs for feature engineering, ETL, graph construction, machine learning, image processing, scoring
- Abstraction level familiar to data scientists; removes complexity of parallel processing
- Lower level Spark APIs seamlessly exposed
- Easy to build apps plugging in other tools/libraries

https://github.com/trustedanalytics/spark-tk
http://trustedanalytics.github.io/spark-tk/

Spark-tk components

- **Frames**
  - Scalable data frame representation
  - More intuitive than low level HDFS file and Spark RDD/DataFrame/DataSet formats; schema inference
  - APIs to manipulate the data frames for feature engineering and exploration, such as joins and aggregations
  - Run user-defined transformations and filters using distributed processing
  - Input to our models
  - Easy to convert to/from Spark-tk Frames - Spark representations

Spark-tk components

- **Graphs**
  - Scalable graph representation based on Frames for vertices and edges
  - In house distributed algorithms for graph analytics using GraphX
  - Supports importing/exporting to OrientDB's scalable graph databases – visualize, real-time graph querying
  - Store massive graphs

Spark-tk components

- **Machine Learning & Streaming**
  - Time series analysis
  - Recommender systems
  - Topic Modeling
  - Clustering
  - Classification/Regression
  - Image processing
  - Scikit Models
  - Streaming via Scoring engine

How Spark-tk helps

- Brings in support for image analytics to Spark
  - Support for ingesting and processing DICOM images in a distributed environment
  - Queries, filters, and analytics on image collections
  - Integrated & distributed **dcm4che3** via. Spark
  - Tested against **pydicom**
  - Used matplotlib for visualization
- Machine learning for image and health records
  - Created distributed Cox Survival Proportional Hazards Model to compare sets of populations
  - Recommend algorithm and hyper-parameters

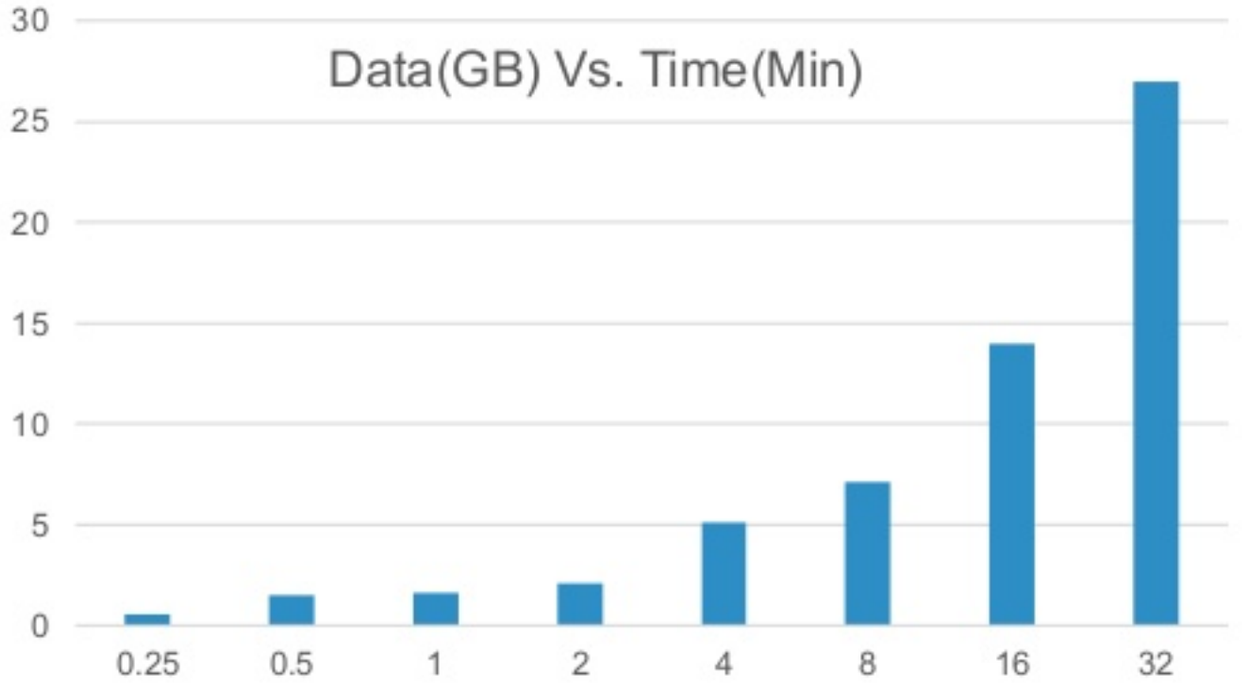
Live Demo

spark-tensorflow-connector

- Developed library for loading and storing TensorFlow records with Apache Spark
- Implements data import from the standard TensorFlow record format (TFRecords) into Spark SQL DataFrames, and data export from DataFrames to TensorFlow records

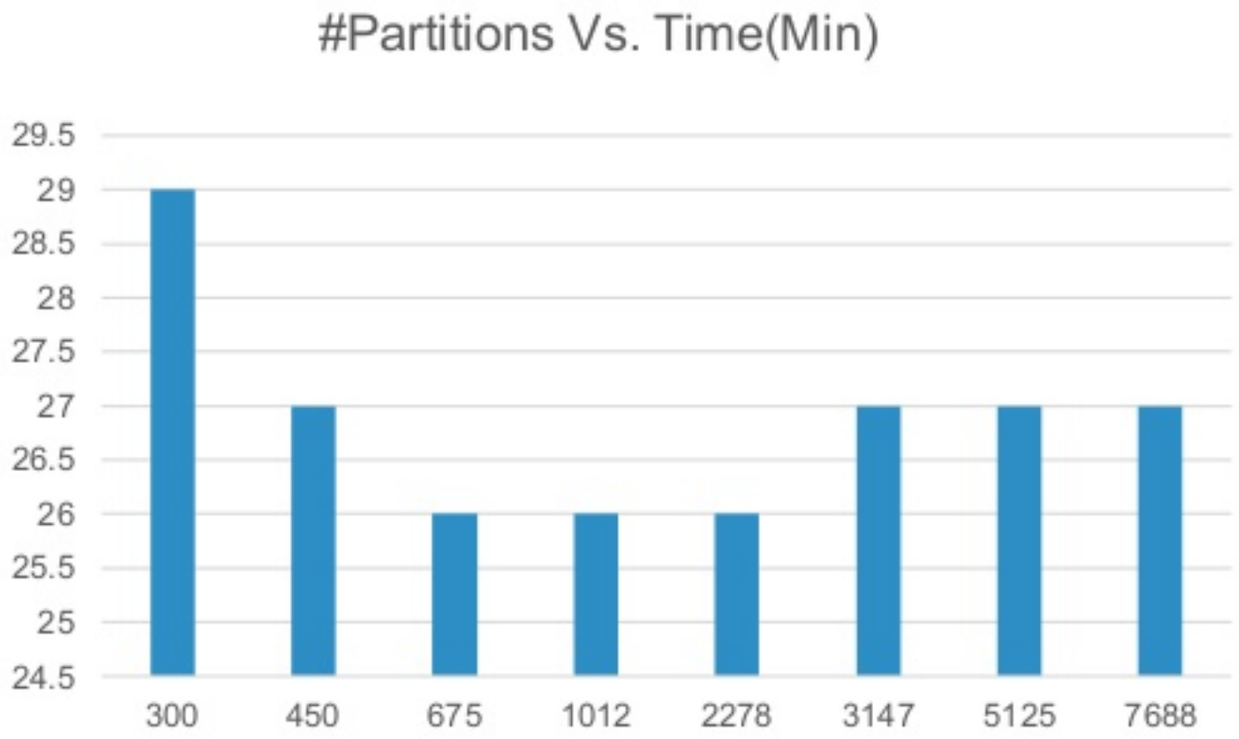
https://github.com/tensorflow/ecosystem/tree/master/spark/spark-tensorflow-connector

Performance



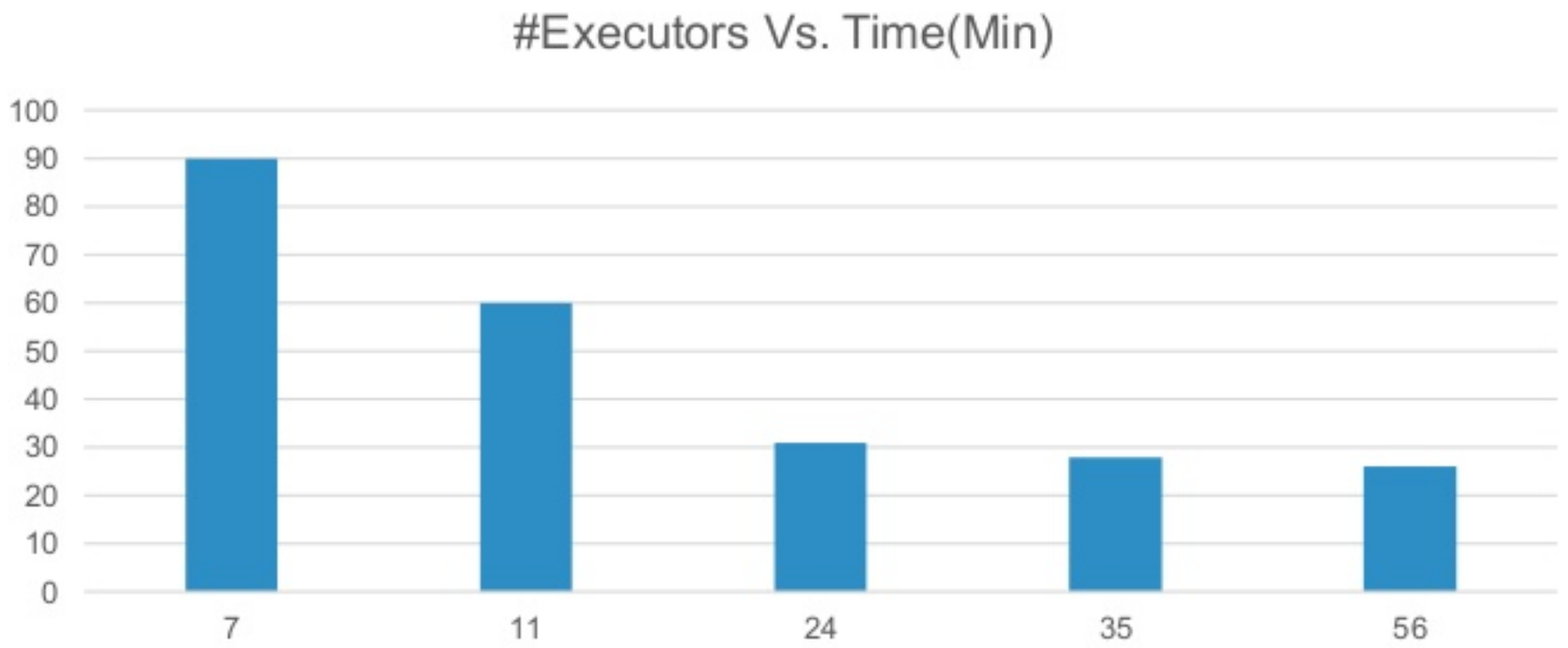
Processing time increases with data(Constant #Partitions=1000)

Performance



Increasing partitions reduced the time and then increased time(overhead)

Performance



Increasing number of executors reduces the time significantly

Performance

Increasing executor cores did not significantly change time

Thank You.

https://github.com/trustedanalytics/spark-tk

http://trustedanalytics.github.io/spark-tk/