Investigating effective use of Deep Learning at KEKCC and future perspective

Wataru Takase
Computing Research Center, KEK

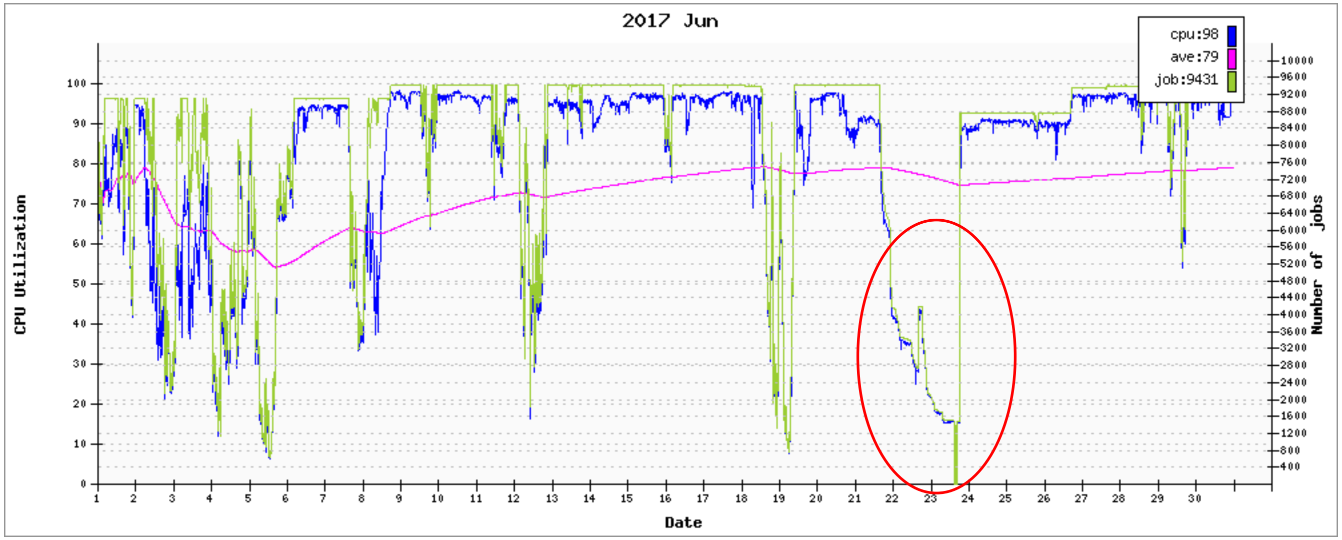
14th February, 2018

Why are we investigating Deep Learning?

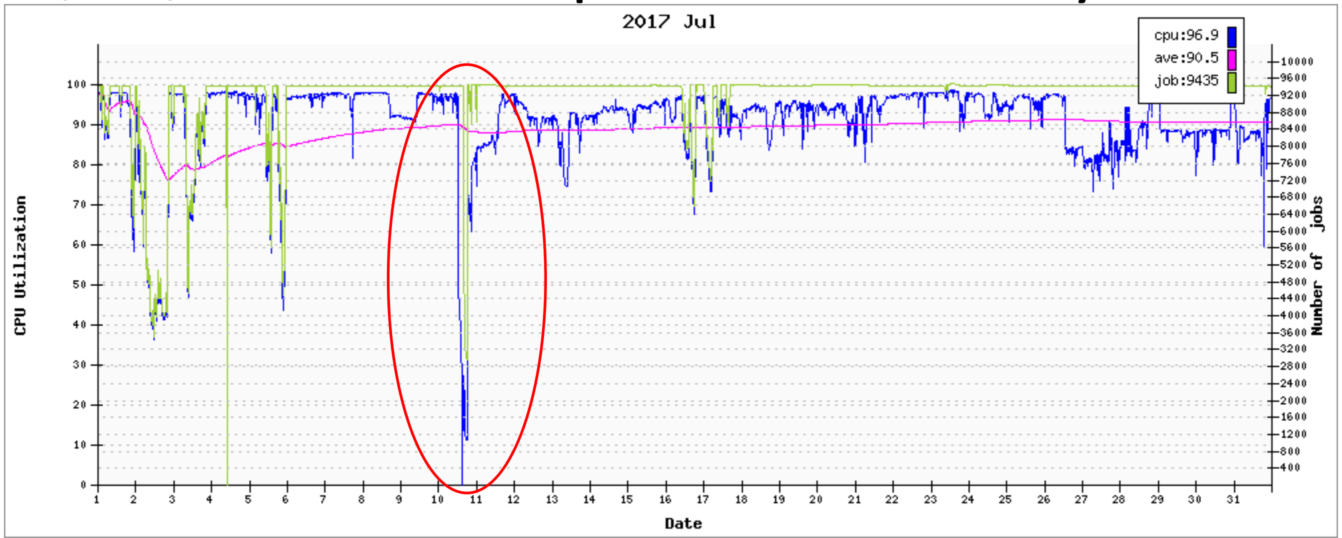
- So popular and seems to be interesting.
- Want to apply it to anomaly detection/prediction for KEKCC.
- Explore the way of applying to the other things.

Examples of Anomaly in KEKCC (1/2)

- 2017/06/23: LSF daemon repeated restarting.

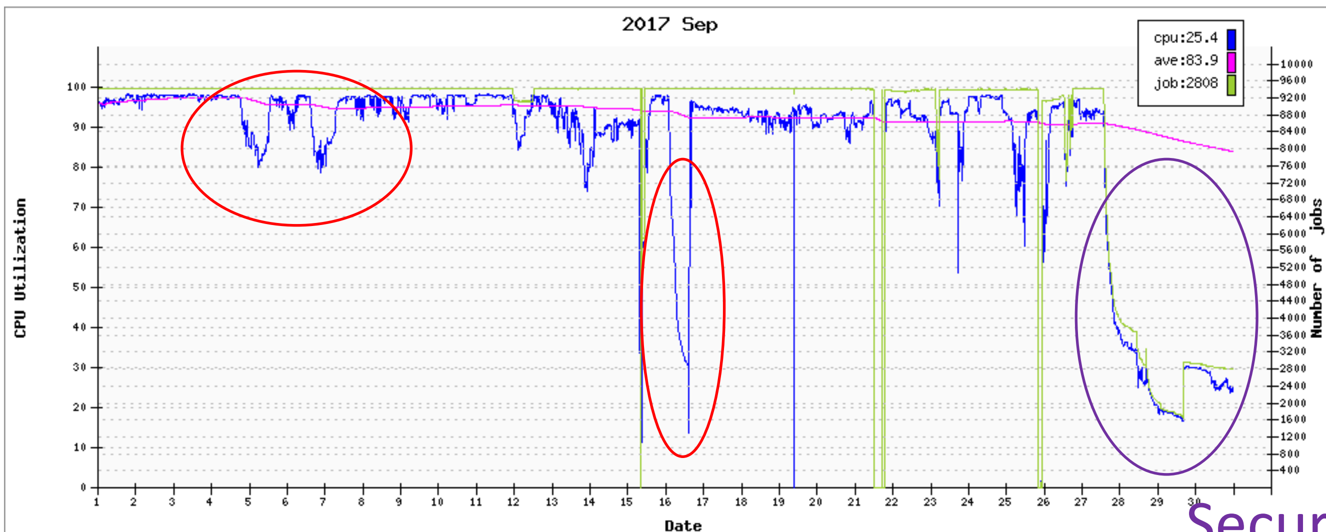


- 2017/07/10: GPFS response were delayed.



Examples of Anomaly in KEKCC (2/2)

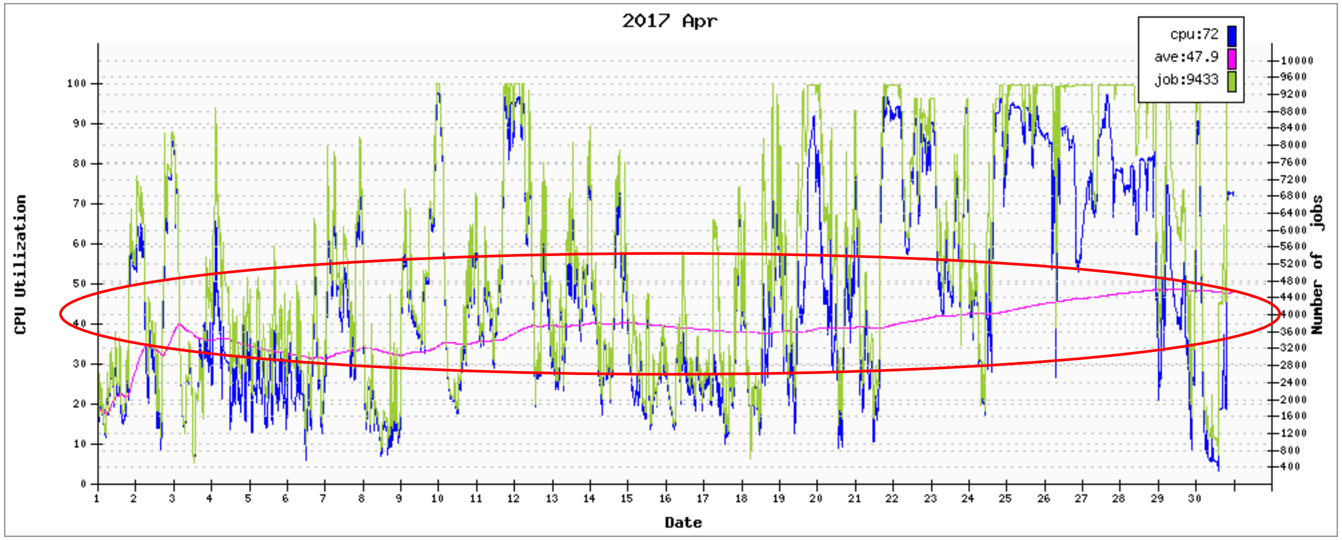
- 2017/09/05,06: Some jobs waited to stage much data from tapes.
- 2017/09/16: GPFS hanged.



Security update

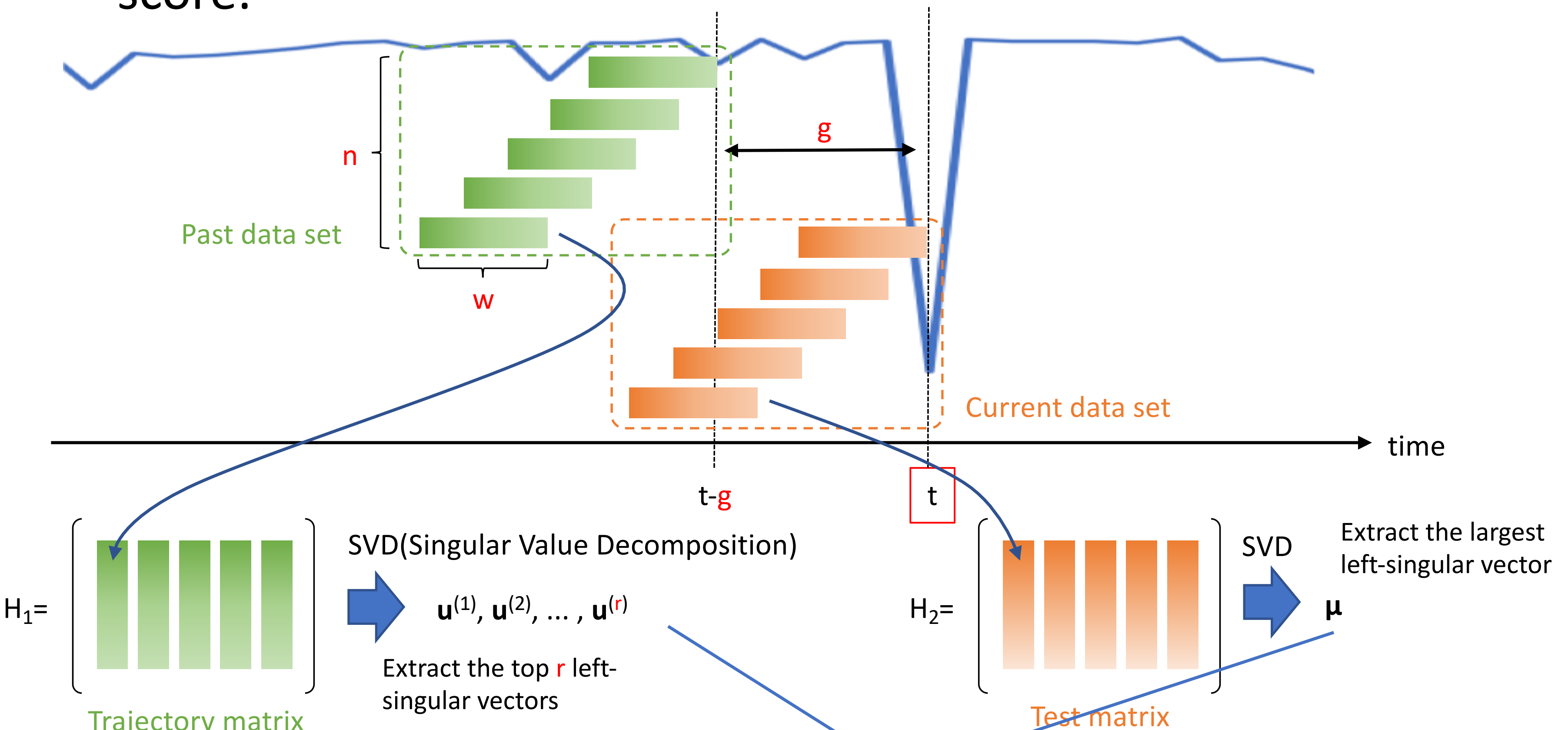
How to detect anomaly: Threshold

- Example: If averaged CPU utilization becomes less than 50%, we suppose anomaly occurred.
- Although averaged CPU utilization was less than 50% all the time in last April, there were no specific anomalies.
  - We guess it was because April is the season that many people arrive/leave.



How to detect anomaly: Change Point Detection

- Example: SST (Singular Spectrum Transformation)[1]
  - Compare current and past dataset and calculate change point score.



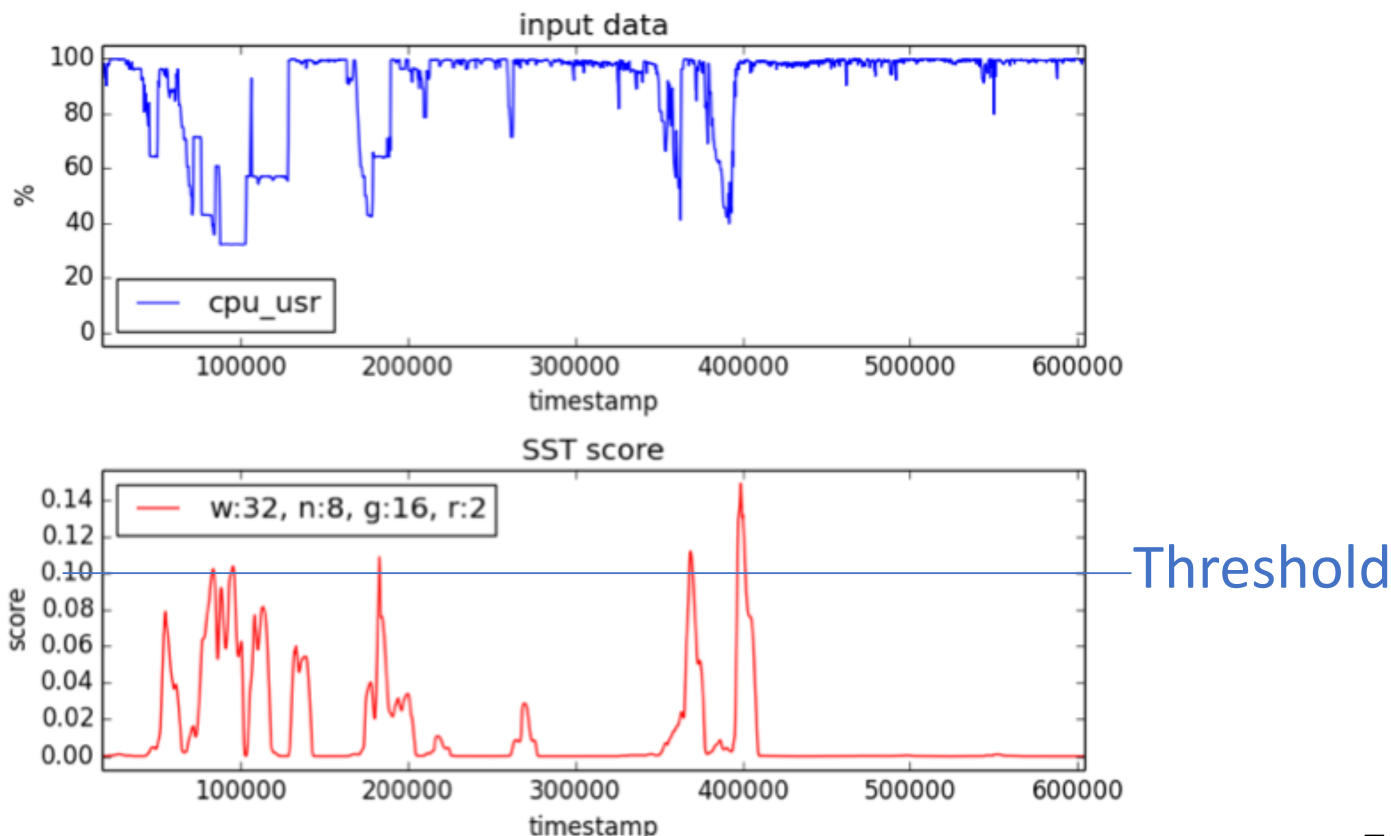
Change point score

$$z(t) = 1 - \sum_{i=1}^{r} \{\mu^T \cdot u^{(i)}\}^2$$

[1] http://ide-research.net/papers/2005_SDM_Ide.pdf

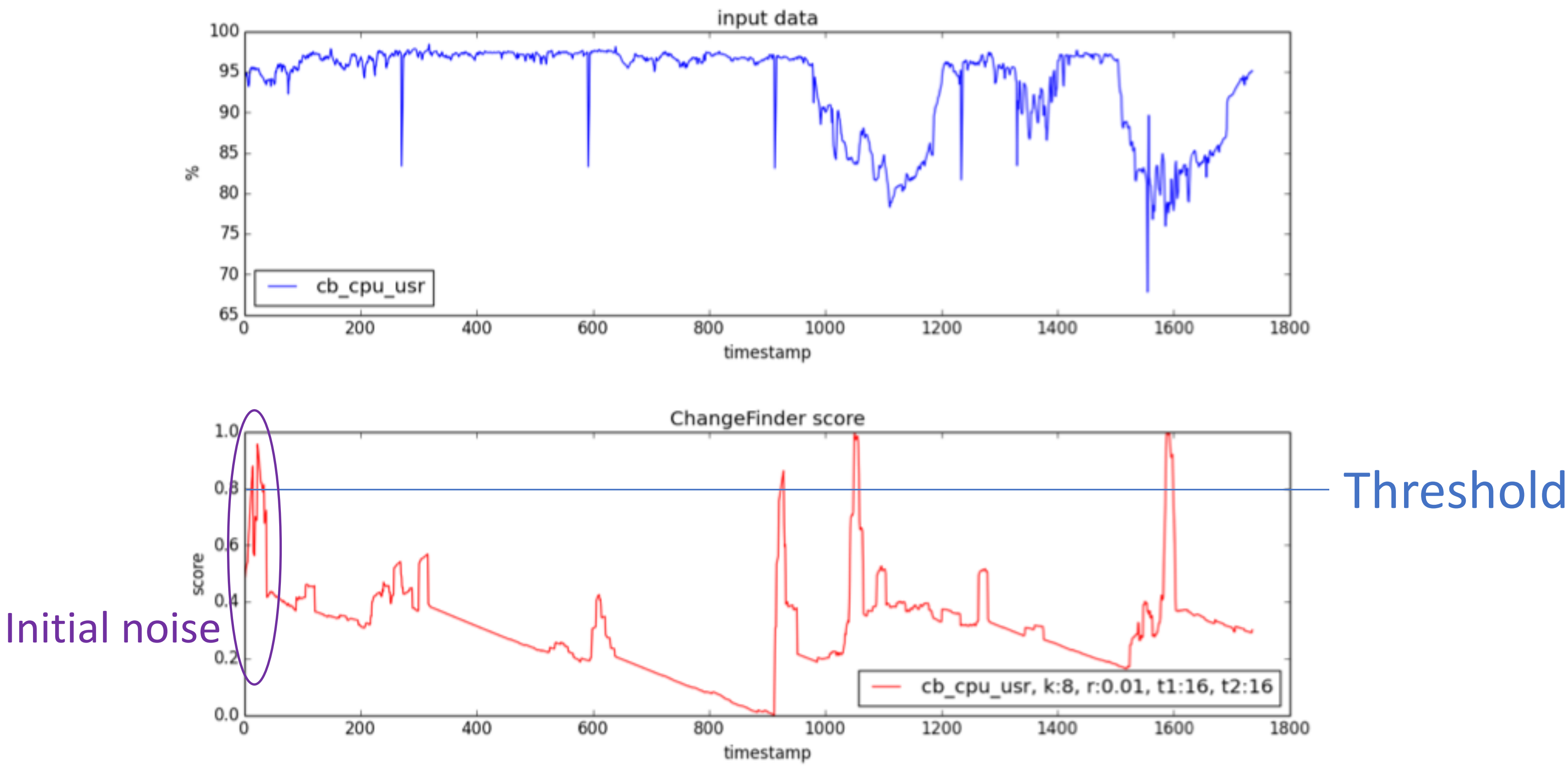
How detect anomaly: Change Point Detection

- We applied SST to CPU utilization of KEKCC for testing.
- If change point scores higher than 0.10, for example, we suppose anomaly occurred.



How detect anomaly: Change Point Detection

- Another algorithm: ChangeFinder[2]



[2] J. Takeuchi and K. Yamanishi, "A unifying framework for detecting outliers and change points from time series"

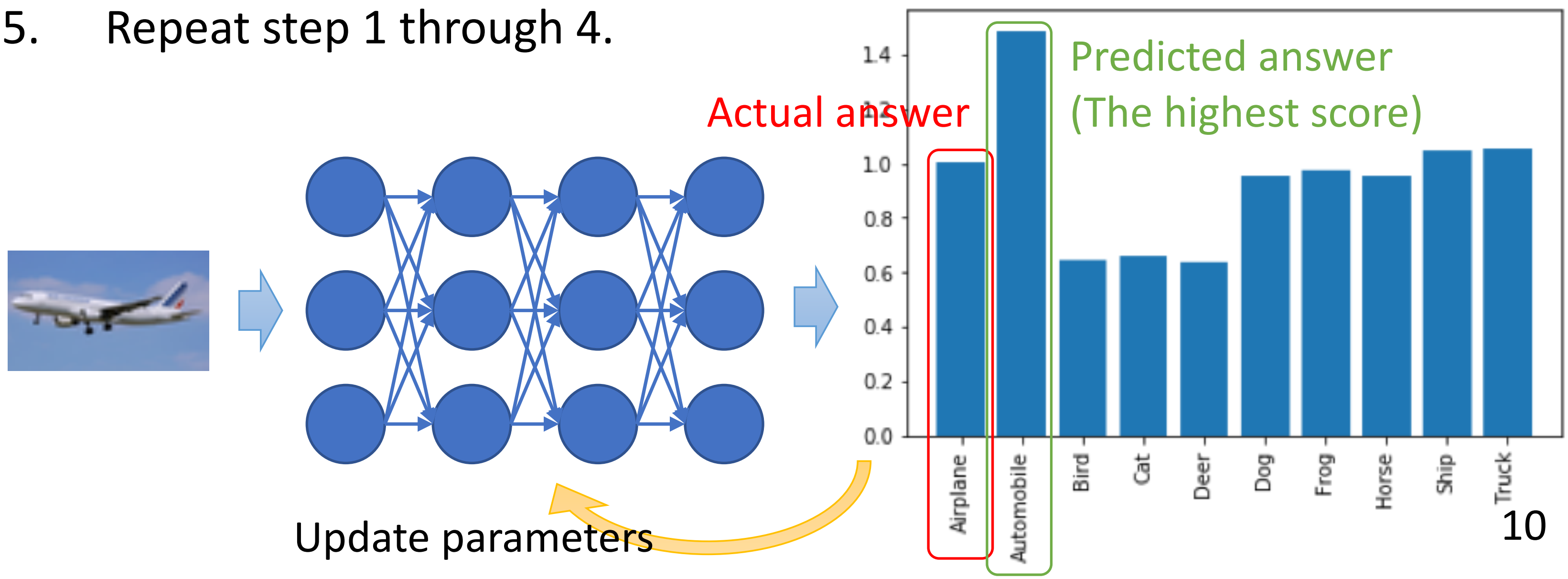
Ways of detecting anomaly

- Threshold detects low service usage as an anomaly.
  - This method sometimes makes misdetection.
- Change Point Detection detects a drastic change in service usage as an anomaly.
- Deep Learning detects an anomaly as an anomaly.
  - Deep Learning becomes to know what state is the anomaly and what is the cause based on large amount of data.
  - It can also be applied to predict anomalies.

What is Deep Learning?

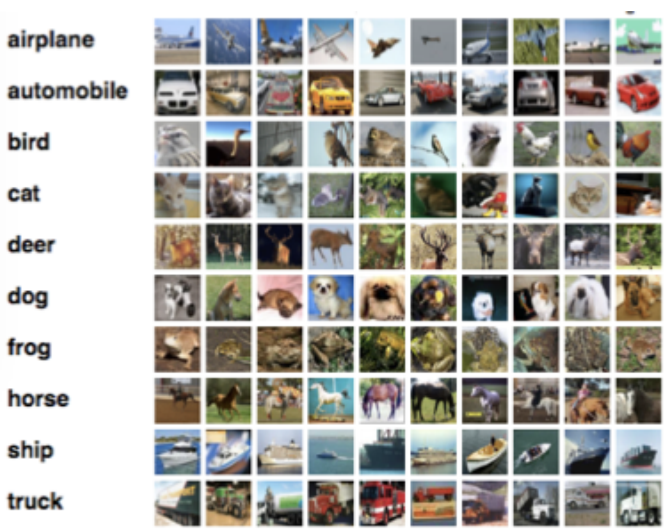
- Subset of Machine Learning.
- Multi-layer Neural Network.
- Use Backpropagation for training algorithm.
- Workflow:

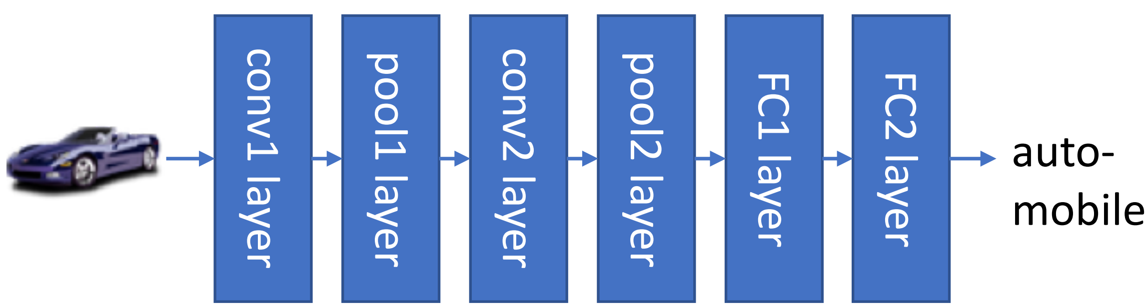
1. Feed samples for training.
2. Calculate neural network output, which means predicting the answer.
3. Calculate loss (the difference between the predicted and the actual answers).
4. Adjust the network parameters to minimize the loss.
5. Repeat step 1 through 4.



Example: CIFAR-10 Image[3] Classification

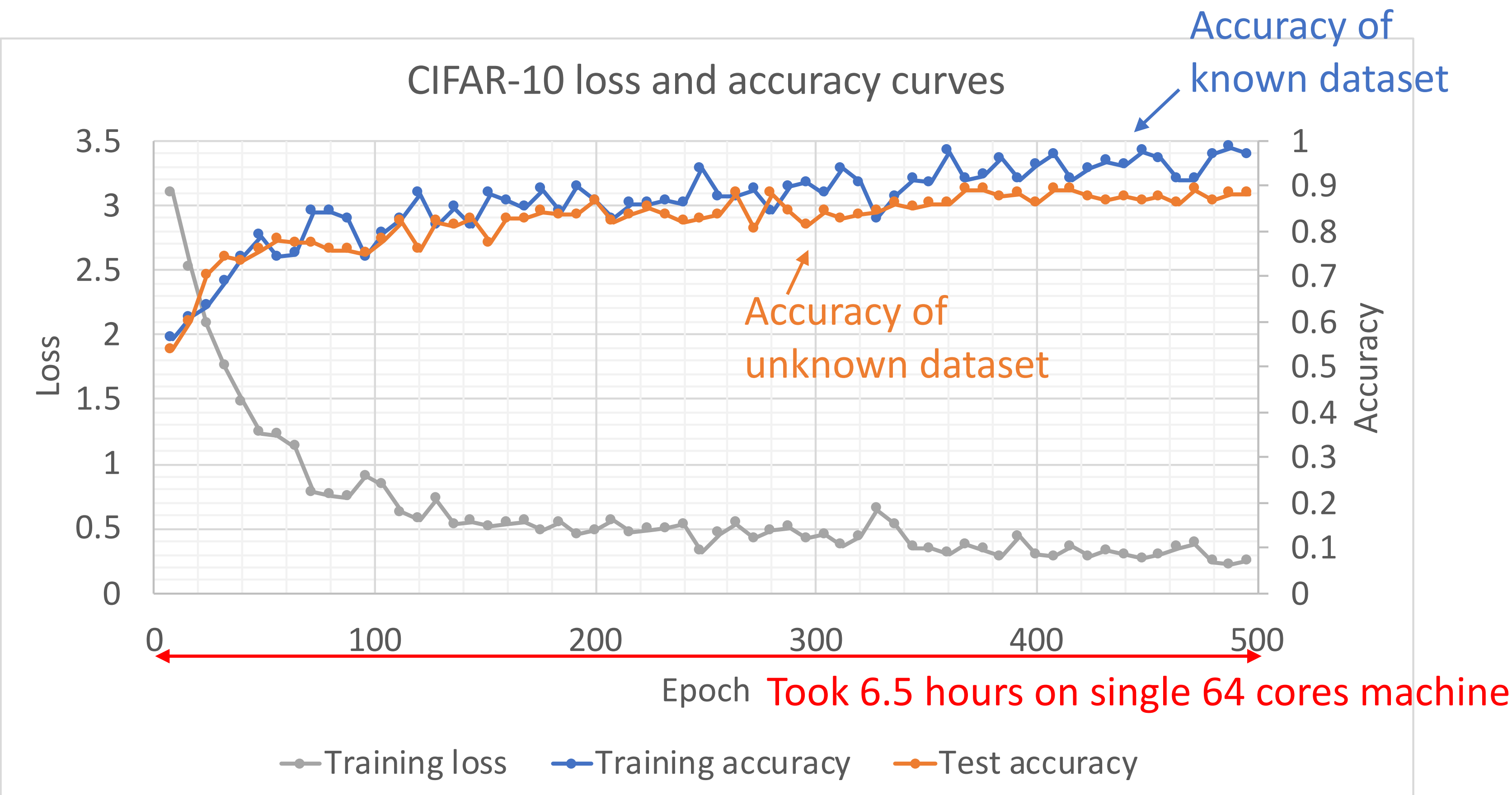
- Classify input image into 10 categories.
  - Although image classification is not our goal, this is good for learning about Deep Learning.
- We have built Convolutional Neural Network, then trained for the recognition using TensorFlow[4].





[3] https://www.cs.toronto.edu/~kriz/cifar.html [4] https://www.tensorflow.org/tutorials/deep_cnn

CIFAR-10 Loss and Accuracy curves



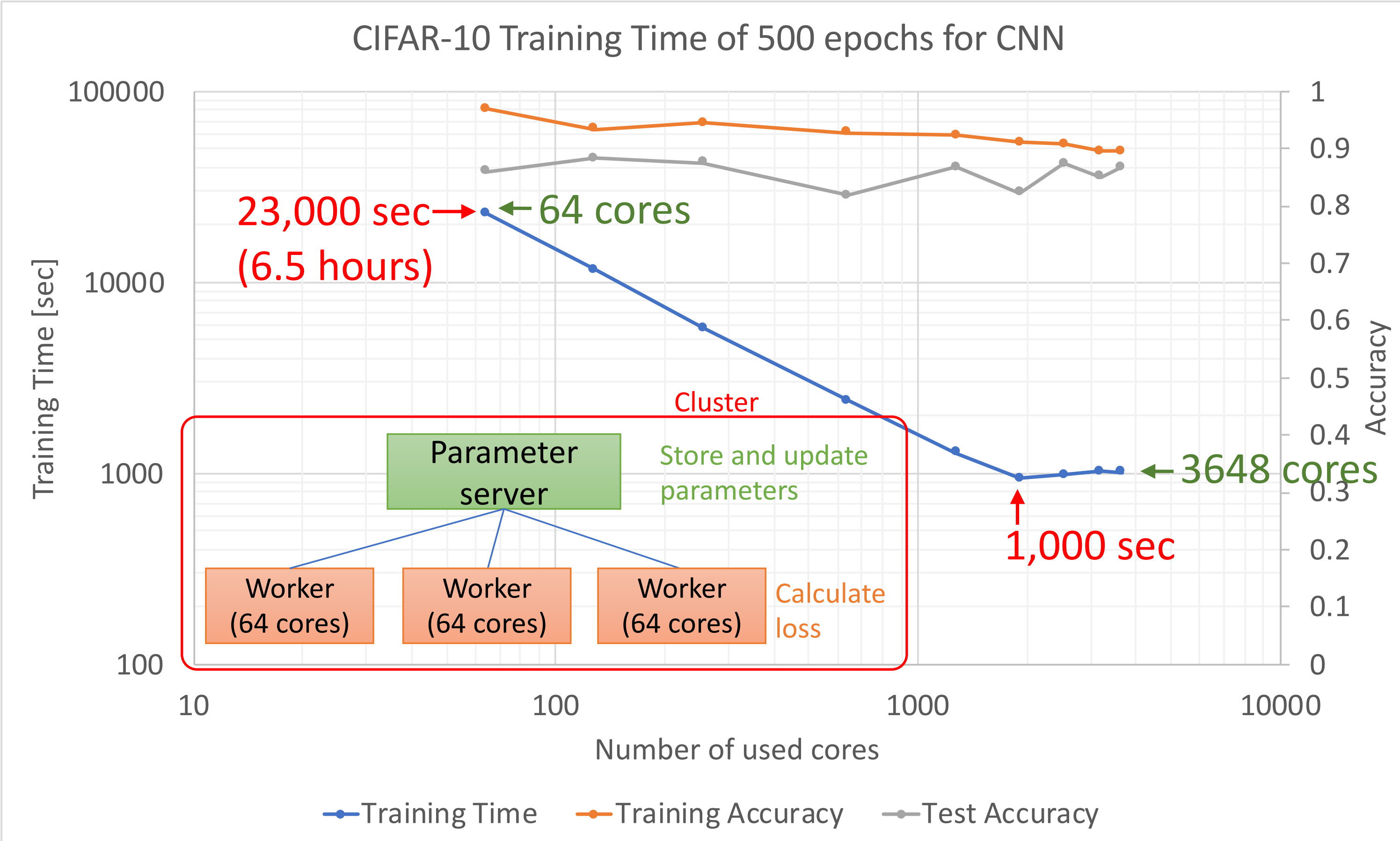
- Repeated training for minimizing the loss (the difference between the network output and the actual answer) against training data set.
- Measured accuracies of training data and test data respectively.
  - Check overfitting happend

Speeding up Deep Learning: GPU

- Many frameworks support CUDA.

	Creator	Interface	Written in	CUDA support	GitHub stars	Note
Theano	University of Montreal, CA	Python	Python	Yes	7,347	• Flexible
Deeplearning4j	Skymind engineering team, US	Java, Scala, Clojure	C, C++	Yes	7,759	• Cooperate with Spark • Support multi-node execution
Caffe	Berkeley Vision and Learning Center, US	Python, C++, MATLAB	C++	Yes	21,474	• Good for image processing • Easy to use
Torch	Ronan Collobert, Koray Kavukcuoglu, Clement Farabet	C, C++, Lua, LuaJIT	C, Lua	Yes	7,482	• Many expansions
TensorFlow	Google Brain team	Python, C	C++, Python	Yes	79,754	• Flexible • Sufficient documents • Support multi-node execution
Chainer	Preferred Networks, JP	Python	Python	Yes	3,181	• Support multi-node execution
CNTK	Microsoft Research	Python, C++	C++	Yes	13,200	• Support multi-node execution • Faster than TensorFlow
MXNet	University of Washington, Carnegie Mellon University, et al.	Python, C++, R, Java, Scala, MATLAB, JavaScript	C++	Yes	12,206	• Support multi-node execution

Speeding up Deep Learning: Multi-node

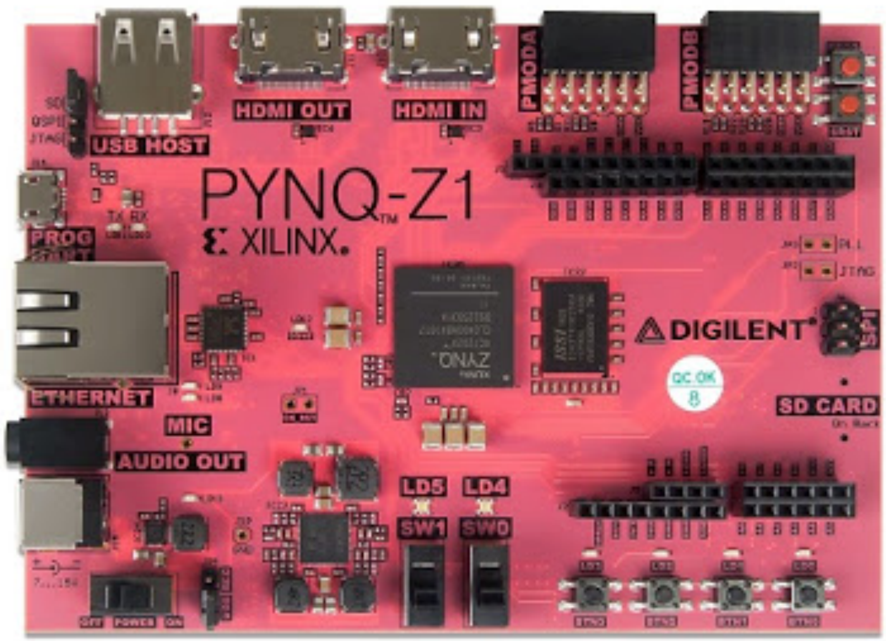


- We have measured scalability of TensorFlow Deep Learning by using m4.16xlarge (64 CPU cores) instances on AWS*.

* Cloud resources used in this work was provided in the Demonstration Experiment of Cloud Use conducted by National Institute of Informatics (NII) Japan (FY2017).

Speeding up Deep Learning: FPGA

- Example: PYNQ-Z1 board[5] (I have one.)
  - Software runs on the ARM CPU.
  - FPGA logics can be invoked by bundled Python library easily.
- You can create own Neural Network model on FPGA and invoke it from Python.
- Example: BNN-PYNQ[6]: Pre-build Binalized Neural Network running on FPGA for PYNQ.



[5] https://www.xilinx.com/products/boards-and-kits/1-hydd4z.html

[6] https://github.com/Xilinx/BNN-PYNQ

CIFAR-10 inference example provided by BNN-PYNQ project

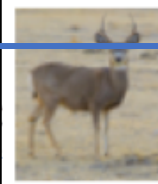
```python
import bnn
from IPython.display import display
from PIL import Image

im = Image.open('/home/xilinx/jupyter_notebooks/bnn/deer.jpg')
im.thumbnail((64, 64), Image.ANTIALIAS)
display(im)

print("# Launching BNN in hardware")
hw_classifier = bnn.CnvClassifier('cifar10')
class_out = hw_classifier.classify_image(im)
print("Class number: {0}".format(class_out))
print("Class name: {0}".format(hw_classifier.class_name(class_out)))

print("# Launching BNN in software")
sw_classifier = bnn.CnvClassifier("cifar10", bnn.RUNTIME_SW)
class_out = sw_classifier.classify_image(im)
print("Class number: {0}".format(class_out))
print("Class name: {0}".format(sw_classifier.c
```

Inference on FPGA by trained network for CIFAR-10

Inference on CPU

↑ PYNQ-Z1 board provides Jupyter Notebook interface.

Output →



x500~ faster

```
# Launching BNN in hardware
Inference took 1588.00 microseconds
Classification rate: 629.72 images per second
Class number: 4
Class name: Deer
# Launching BNN in software
Inference took 820699.00 microseconds
Classification rate: 1.22 images per second
Class number: 4
Class name: Deer
```

Summary

- Anomaly detection/prediction is necessary for providing stable IT service.
- Deep Learning may help it and could be applied to the other fields.
- We have been investigating about Deep Learning and the ways of speeding up.
  - The next step is to consider how to apply it to anomaly detection.
- If you are interested in, we welcome your collaboration!!