

# Benchmark on Deep Learning Frameworks

김형준

# ARTIFICIAL INTELLIGENCE

Early artificial intelligence stirs excitement.



# MACHINE LEARNING

Machine learning begins to flourish.



# DEEP LEARNING

Deep learning breakthroughs drive AI boom.



1950's

1960's

1970's

1980's

1990's

2000's

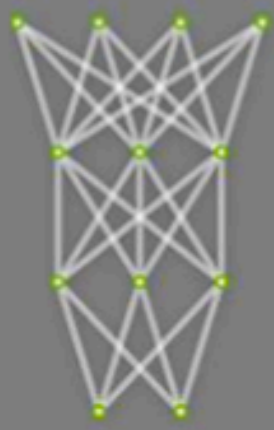
2010's

# DEEP LEARNING

## TRAINING

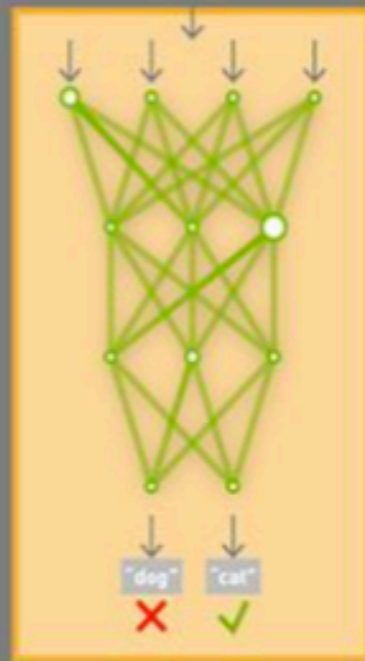
Learning a new capability  
from existing data

Untrained  
Neural Network  
Model



Deep Learning  
Framework

TRAINING  
DATASET



Trained Model  
New Capability



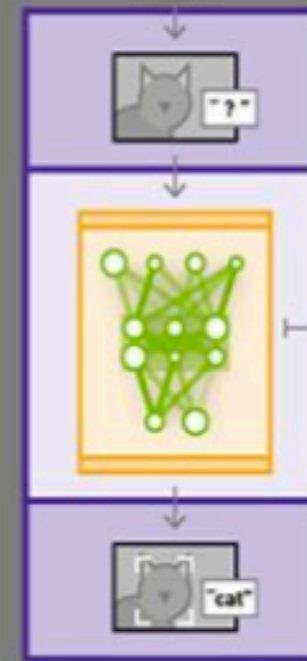
## INFERENCE

Applying this capability  
to new data

NEW  
DATA



App or Service  
Featuring Capability



Trained Model  
Optimized for  
Performance

imperative  
symbolic

theano



before

2012

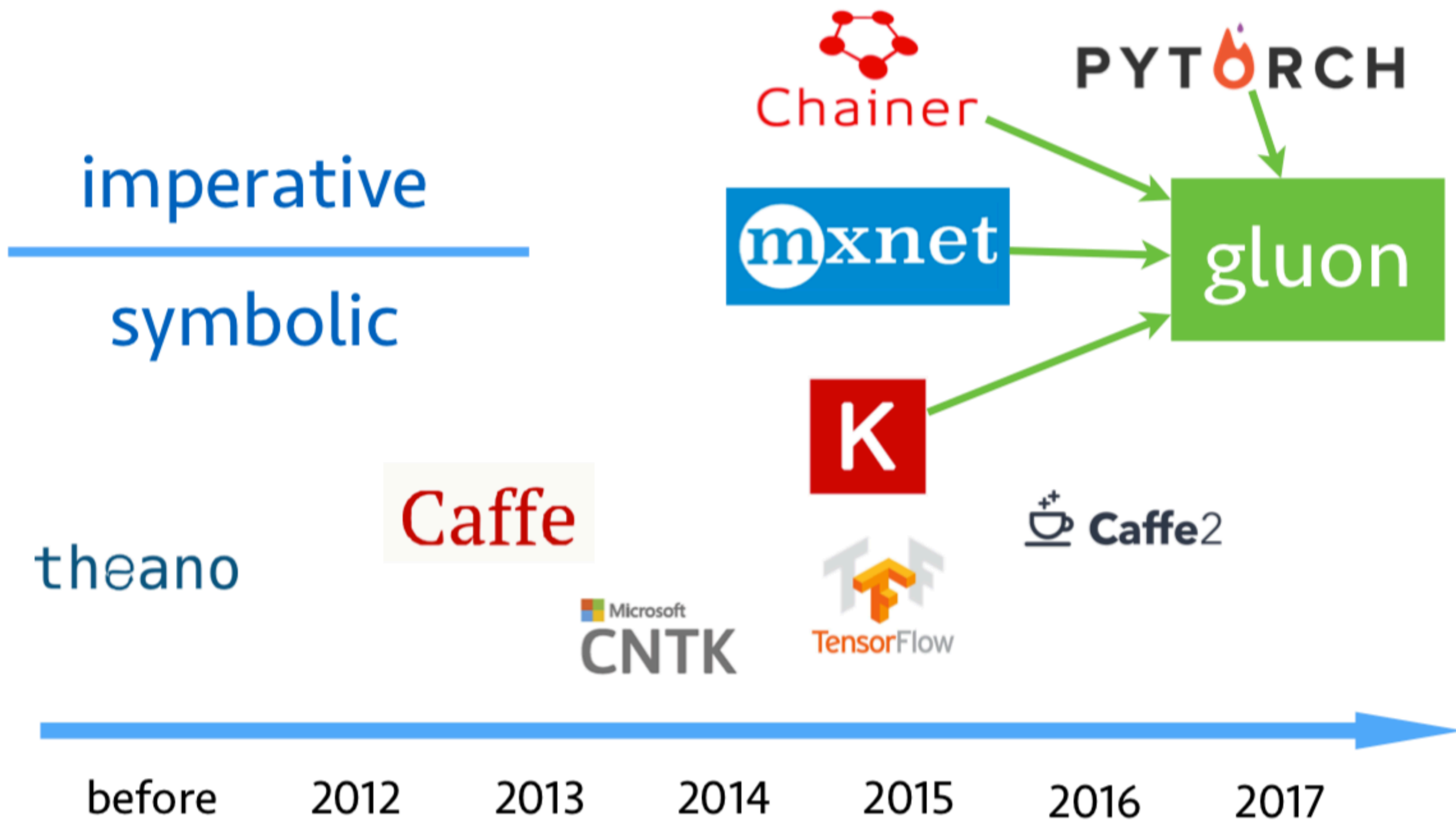
2013

2014

2015

2016

2017



**빨강 파랑 노랑**

파랑 노랑 빨강

# Framework Benchmark

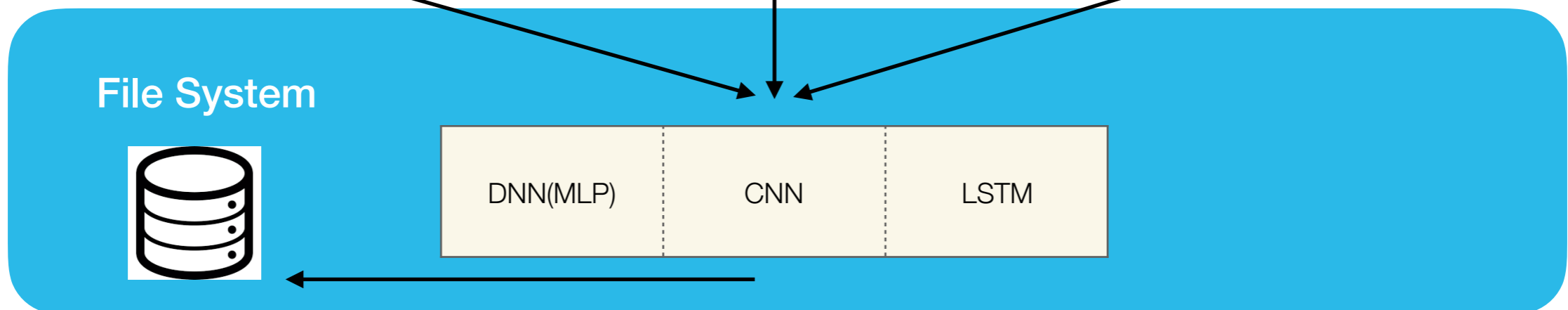
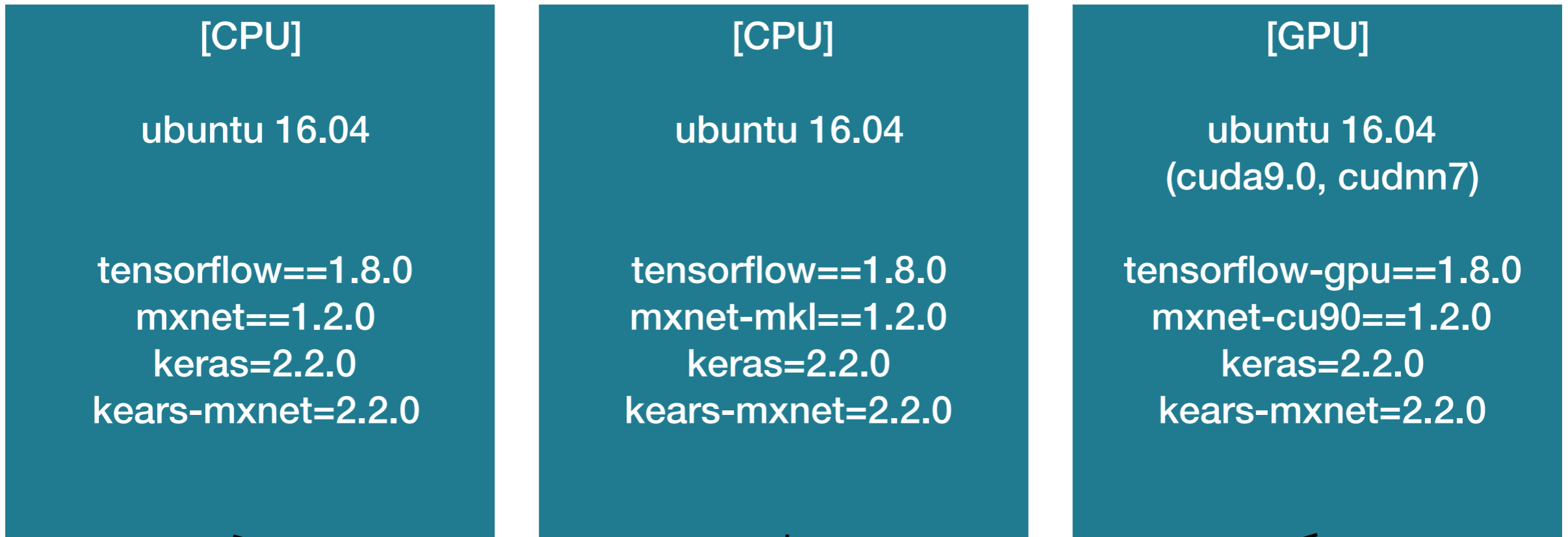
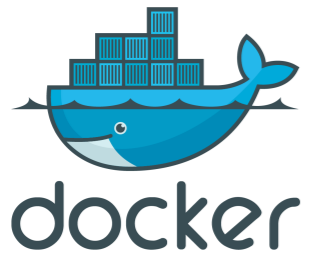
[https://github.com/tensorflow/benchmarks/tree/master/scripts/keras\\_benchmarks](https://github.com/tensorflow/benchmarks/tree/master/scripts/keras_benchmarks)

[https://github.com/aws-labs/keras-apache-mxnet/tree/keras2\\_mxnet\\_backend/benchmark](https://github.com/aws-labs/keras-apache-mxnet/tree/keras2_mxnet_backend/benchmark)

	Tensorflow	CPU (FULL-CORE)	GPU (1-GPU)
D	MXNET	CPU (FULL-CORE)	GPU (1-GPU)
	DNN(MLP)		
	CNN		
	LSTM		
	RN		

**keras-mxnet**

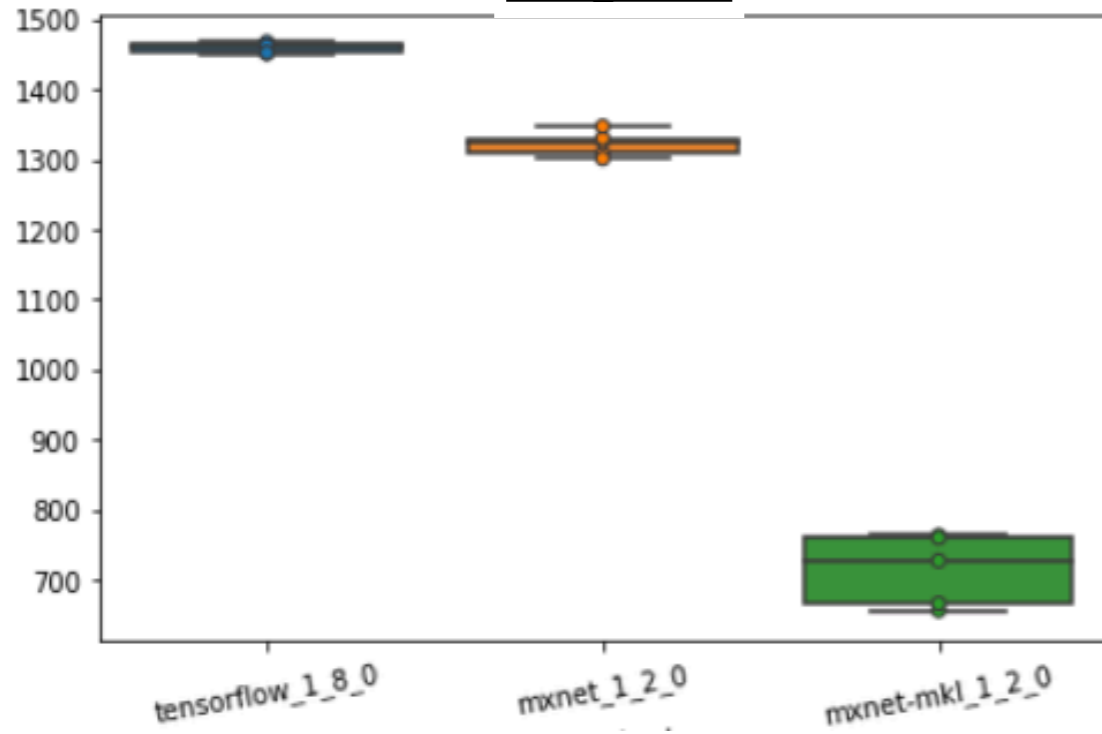




# Between Platform - CPU

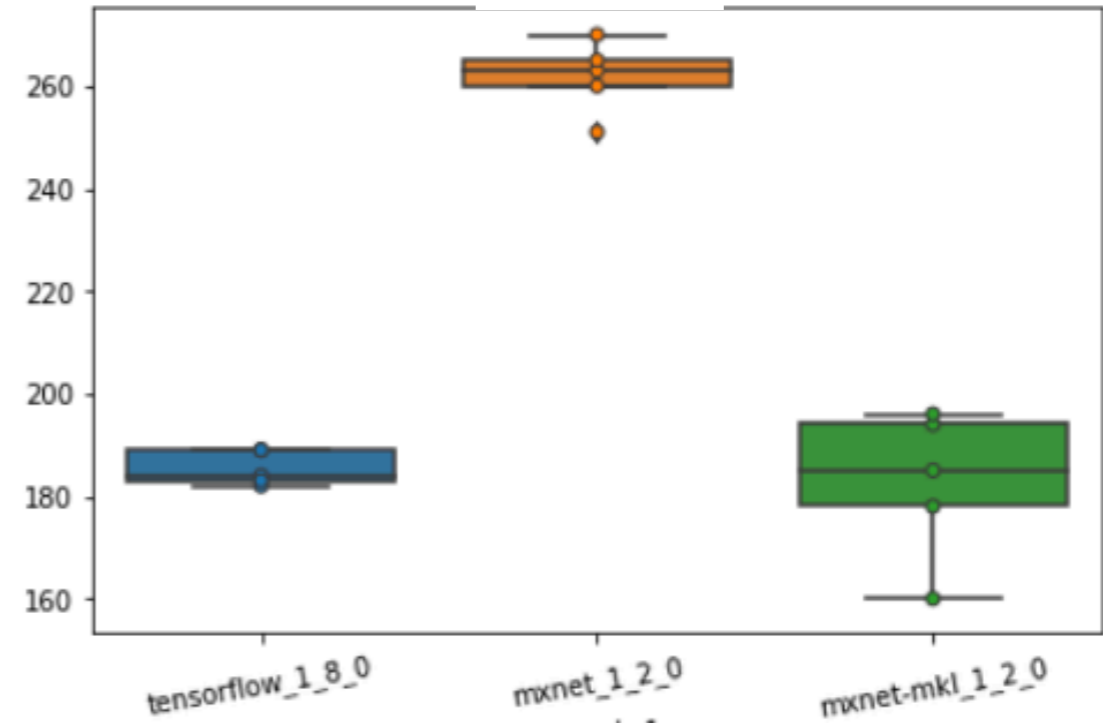
(ms)

*DNN TRAIN*



(ms)

*DNN INFER*



# Tensorflow - CPU

Line #	Hits	Time	Per Hit	% Time	Line Contents
47					@profile
48					def run_benchmark_train(self, gpus=0):
49					
50	1	2.0	2.0	0.0	x_train, y_train = self.x_train, self.y_train
51					
52	1	5569.0	5569.0	0.4	model = Sequential()
53	1	15978.0	15978.0	1.1	model.add(Dense(512, activation='relu', input_shape=(784,)))
54	1	19014.0	19014.0	1.3	model.add(Dropout(0.2))
55	1	14490.0	14490.0	1.0	model.add(Dense(512, activation='relu'))
56	1	17554.0	17554.0	1.2	model.add(Dropout(0.2))
57	1	14042.0	14042.0	1.0	model.add(Dense(self.num_classes))
58					
59	1	3.0	3.0	0.0	if keras.backend.backend() is "tensorflow" and gpus > 1:
60					model = multi_gpu_model(model, gpus=gpus)
61					
62	1	0.0	0.0	0.0	model.compile(loss='categorical_crossentropy',
63	1	10901.0	10901.0	0.8	optimizer=RMSprop(),
64	1	28131.0	28131.0	1.9	metrics=['accuracy'])
65					
66					# create a distributed trainer for cntk
67	1	2.0	2.0	0.0	if keras.backend.backend() is "cntk" and gpus > 1:
68					start, end = cntk_gpu_mode_config(model, x_train.shape[0])
69					x_train = x_train[start: end]
70					y_train = y_train[start: end]
71					
72	1	3.0	3.0	0.0	time_callback = timehistory.TimeHistory()
73					
74	1	2.0	2.0	0.0	model.fit(x_train, y_train, batch_size=self.batch_size,
75	1	1.0	1.0	0.0	epochs=self.epochs, verbose=1,
76	1	1326038.0	1326038.0	91.3	callbacks=[time_callback])
77					
78	1	5.0	5.0	0.0	self.fit_time = 0
79	2	3.0	1.5	0.0	for i in range(1, self.epochs):
80	1	1.0	1.0	0.0	self.fit_time += time_callback.times[i]
81					
82	1	0.0	0.0	0.0	self.model = model

# mxnet-mkl - CPU

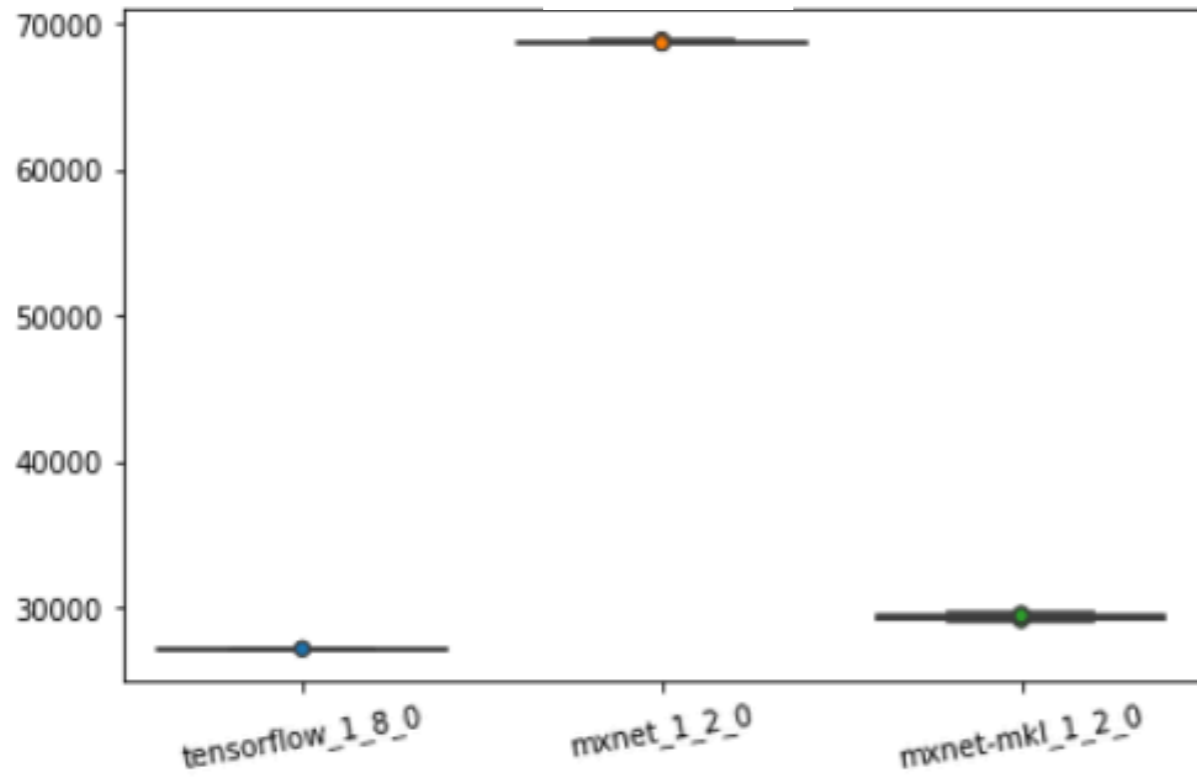
Line #	Hits	Time	Per Hit	% Time	Line Contents
47					@profile
48					def run_benchmark_train(self, gpus=0):
49					
50	1	3.0	3.0	0.0	x_train, y_train = self.x_train, self.y_train
51					
52	1	8234.0	8234.0	1.2	model = Sequential()
53	1	10504.0	10504.0	1.5	model.add(Dense(512, activation='relu', input_shape=(784,)))
54	1	565.0	565.0	0.1	model.add(Dropout(0.2))
55	1	6512.0	6512.0	1.0	model.add(Dense(512, activation='relu'))
56	1	706.0	706.0	0.1	model.add(Dropout(0.2))
57	1	5017.0	5017.0	0.7	model.add(Dense(self.num_classes))
58					
59	1	7.0	7.0	0.0	if keras.backend.backend() is "tensorflow" and gpus > 1:
60					model = multi_gpu_model(model, gpus=gpus)
61					
62	1	1.0	1.0	0.0	model.compile(loss='categorical_crossentropy',
63	1	368.0	368.0	0.1	optimizer=RMSprop(),
64	1	4409.0	4409.0	0.6	metrics=['accuracy'])
65					
66					# create a distributed trainer for cntk
67	1	1.0	1.0	0.0	if keras.backend.backend() is "cntk" and gpus > 1:
68					start, end = cntk_gpu_mode_config(model, x_train.shape[0])
69					x_train = x_train[start: end]
70					y_train = y_train[start: end]
71					
72	1	7.0	7.0	0.0	time_callback = timehistory.TimeHistory()
73					
74	1	2.0	2.0	0.0	model.fit(x_train, y_train, batch_size=self.batch_size,
75	1	0.0	0.0	0.0	epochs=self.epochs, verbose=1,
76	1	643038.0	643038.0	94.7	callbacks=[time_callback])
77					
78	1	4.0	4.0	0.0	self.fit_time = 0
79	2	4.0	2.0	0.0	for i in range(1, self.epochs):
80	1	2.0	2.0	0.0	self.fit_time += time_callback.times[i]
81					
82	1	1.0	1.0	0.0	self.model = model

# mxnet-mkl - CPU

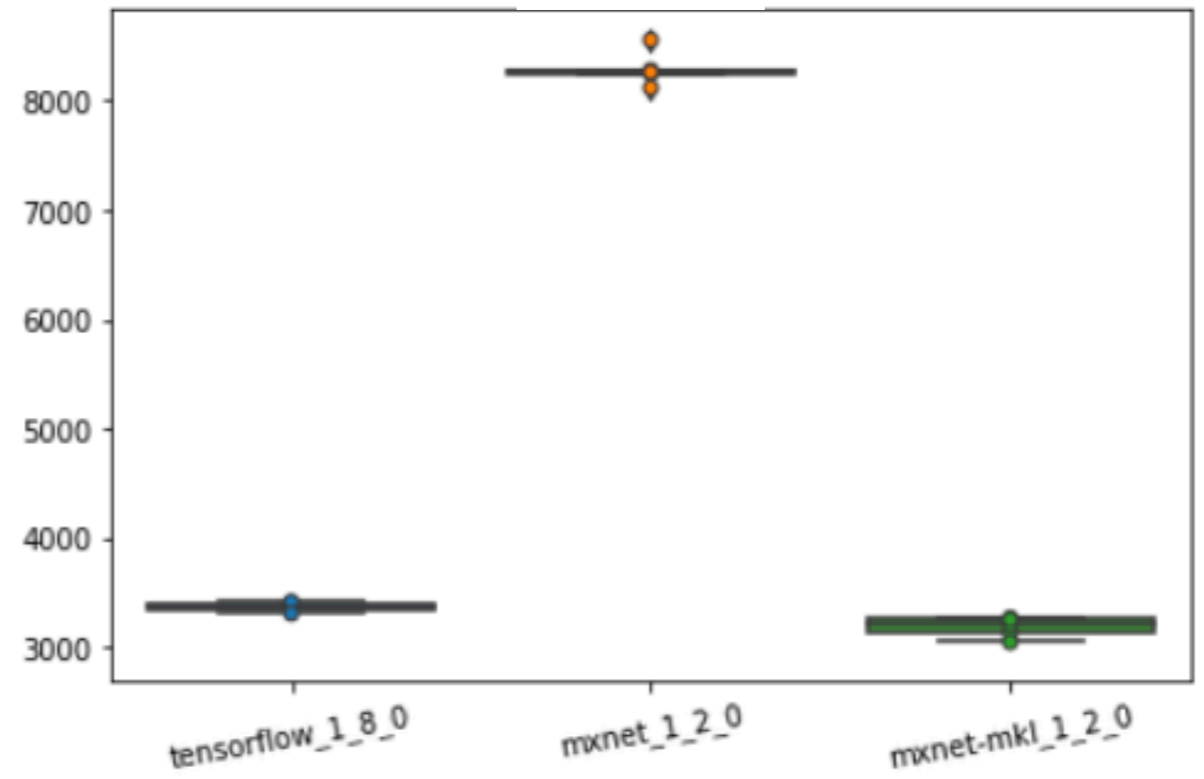
1	[      92.2%]	15	[      100.0%]	29	[      100.0%]	43	[      100.0%]
2	[      100.0%]	16	[      100.0%]	30	[      94.2%]	44	[      100.0%]
3	[      97.4%]	17	[      100.0%]	31	[      100.0%]	45	[      100.0%]
4	[      90.3%]	18	[      100.0%]	32	[      98.1%]	46	[      100.0%]
5	[      94.8%]	19	[      100.0%]	33	[      100.0%]	47	[      100.0%]
6	[      97.4%]	20	[      100.0%]	34	[      100.0%]	48	[      100.0%]
7	[      96.1%]	21	[      100.0%]	35	[      66.2%]	49	[      100.0%]
8	[      100.0%]	22	[      100.0%]	36	[      100.0%]	50	[      100.0%]
9	[      93.5%]	23	[      100.0%]	37	[      100.0%]	51	[      100.0%]
10	[      100.0%]	24	[      100.0%]	38	[      94.2%]	52	[      100.0%]
11	[      100.0%]	25	[      100.0%]	39	[      98.1%]	53	[      99.4%]
12	[      98.1%]	26	[      100.0%]	40	[      100.0%]	54	[      100.0%]
13	[      99.4%]	27	[      100.0%]	41	[      100.0%]	55	[      100.0%]
14	[      98.1%]	28	[      100.0%]	42	[      99.4%]	56	[      98.7%]

Mem [||||| 5.846/7566] Task: 462, 1407, 463, 57, 58, 59, 60, 61, 62, 63, 64, 65, 66, 67, 68, 69, 70, 71, 72, 73, 74, 75, 76, 77, 78, 79, 80, 81, 82, 83, 84, 85, 86, 87, 88, 89, 90, 91, 92, 93, 94, 95, 96, 97, 98, 99, 100

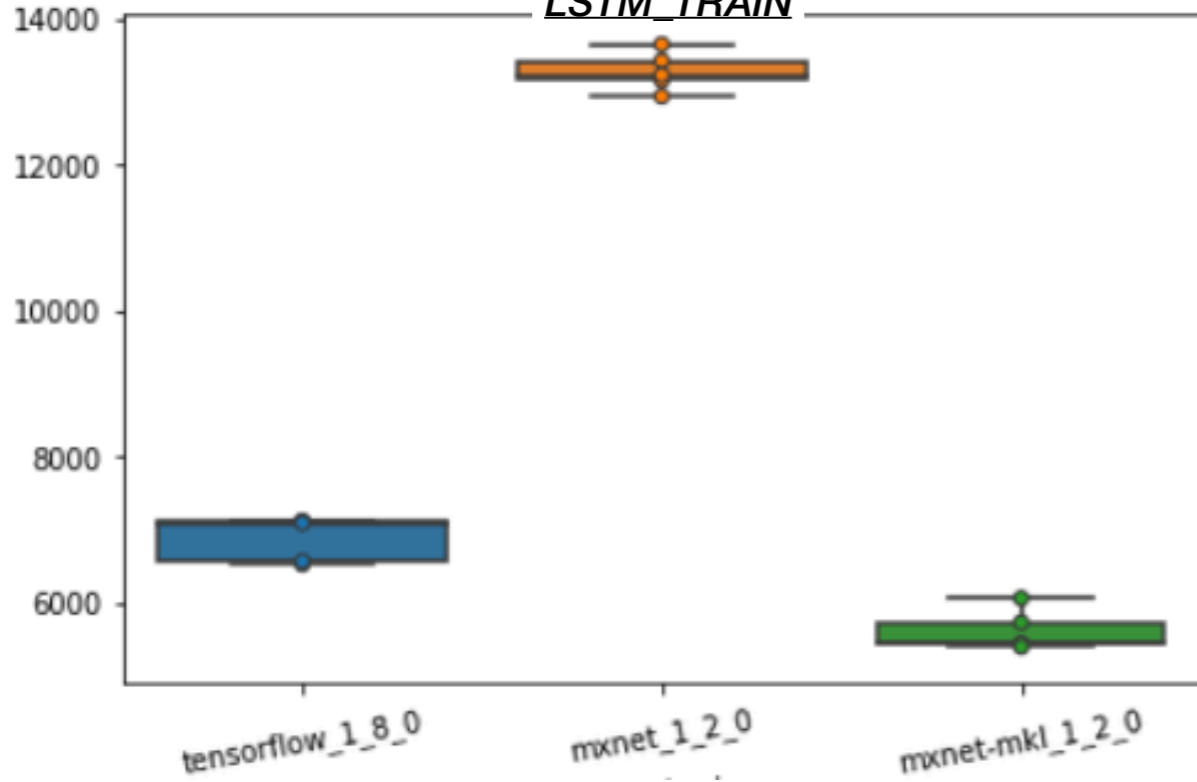
**CNN TRAIN**



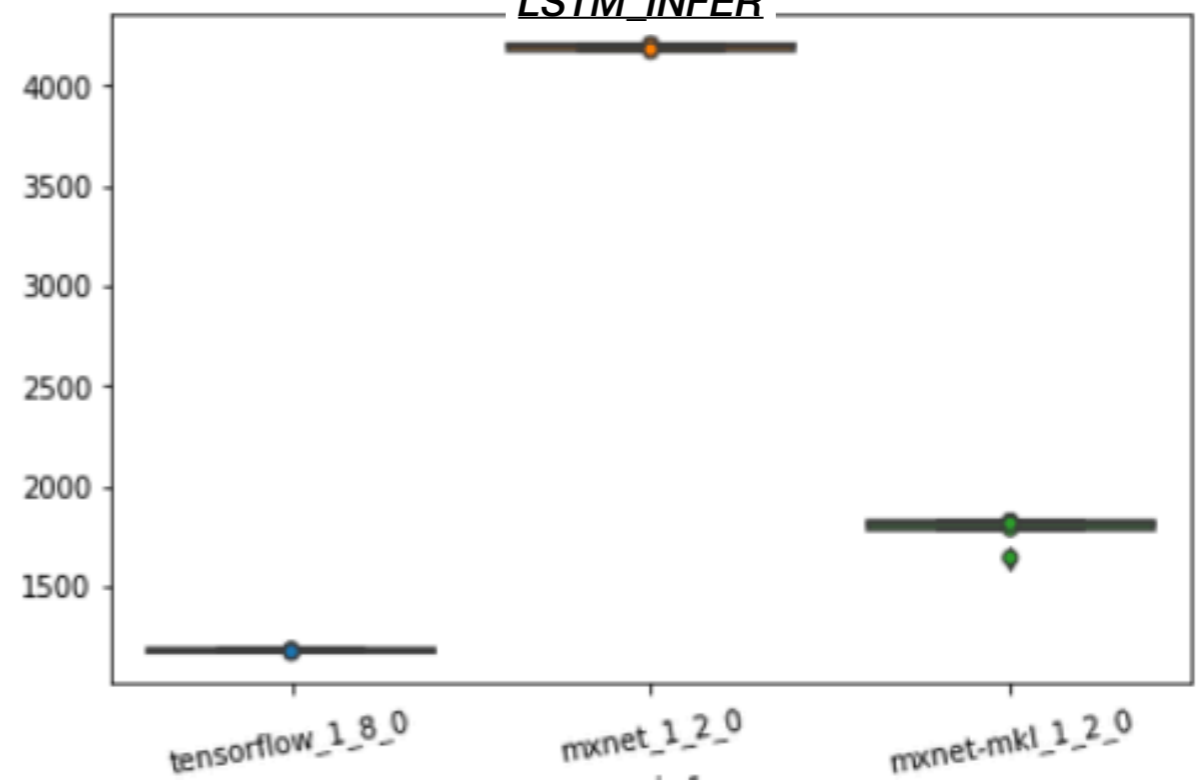
**CNN INFER**



**LSTM TRAIN**

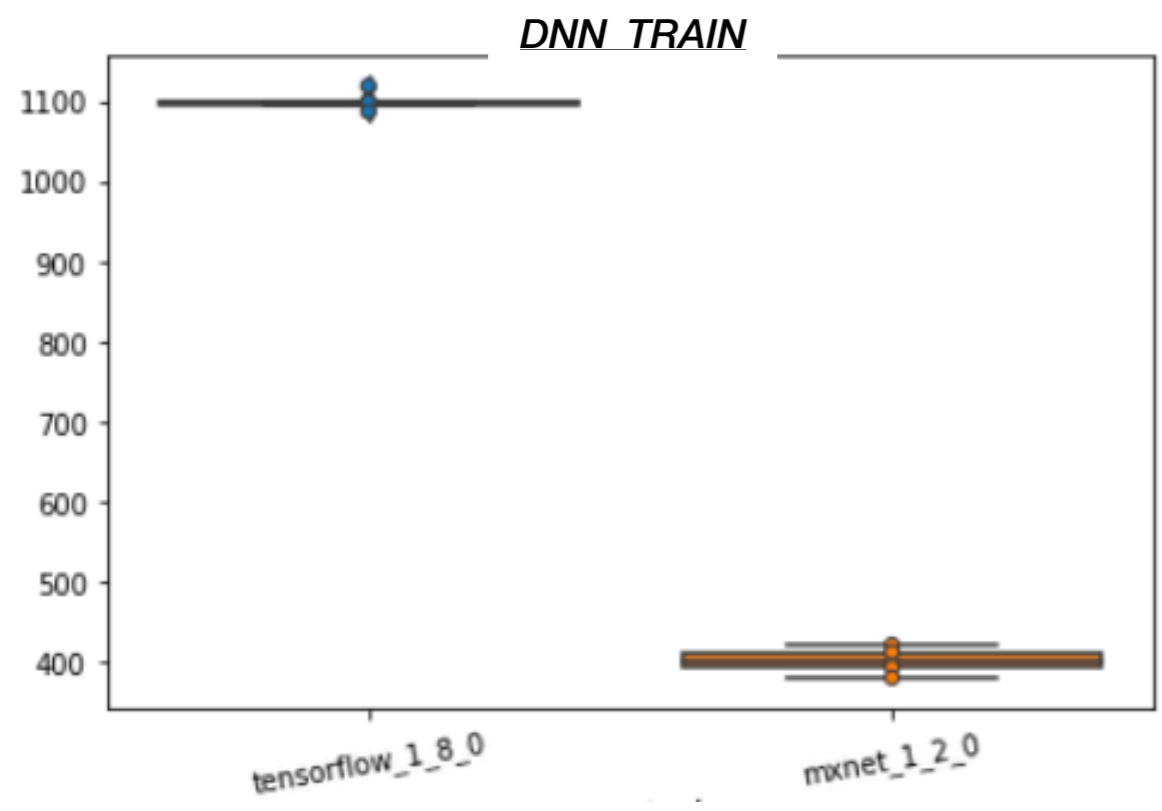


**LSTM INFER**

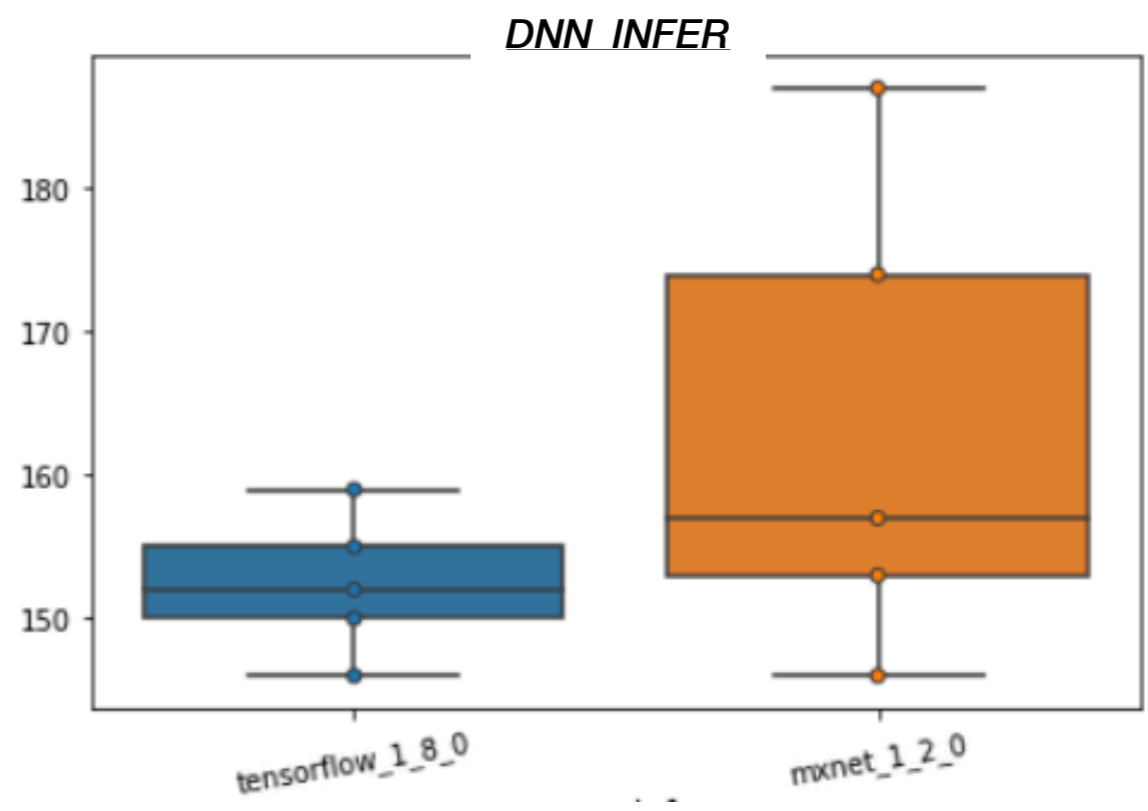


# Between Platform - GPU

(ms)

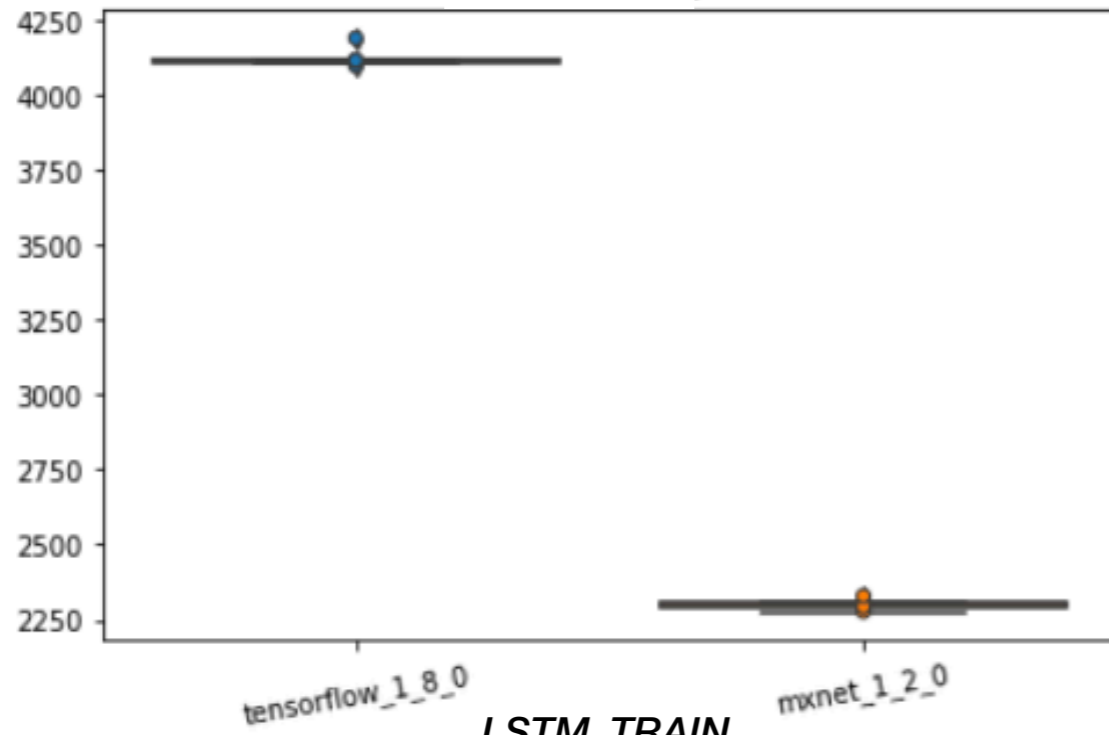


(ms)

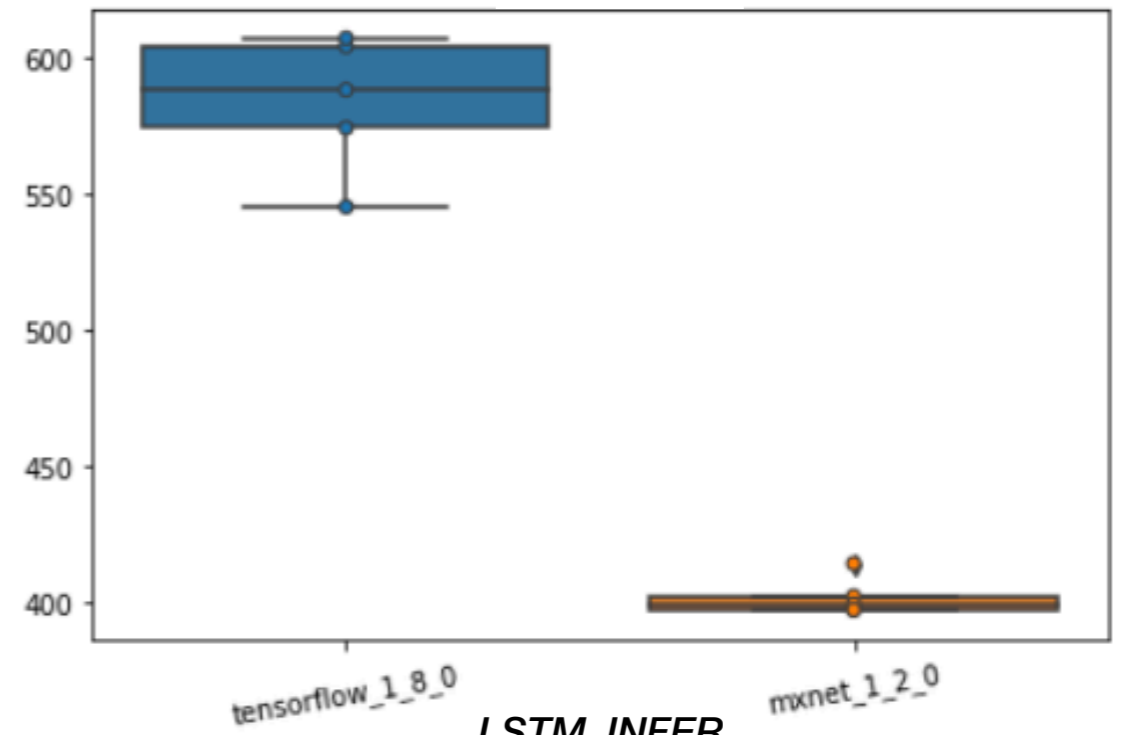


# Between Platform - GPU

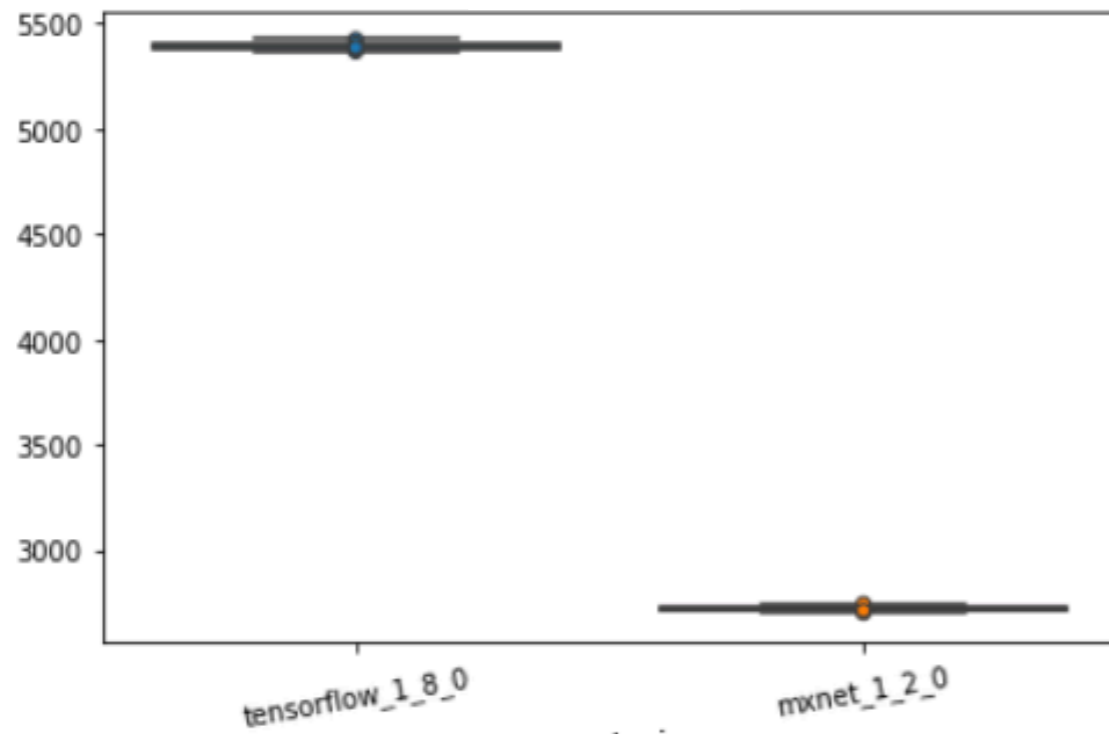
CNN TRAIN



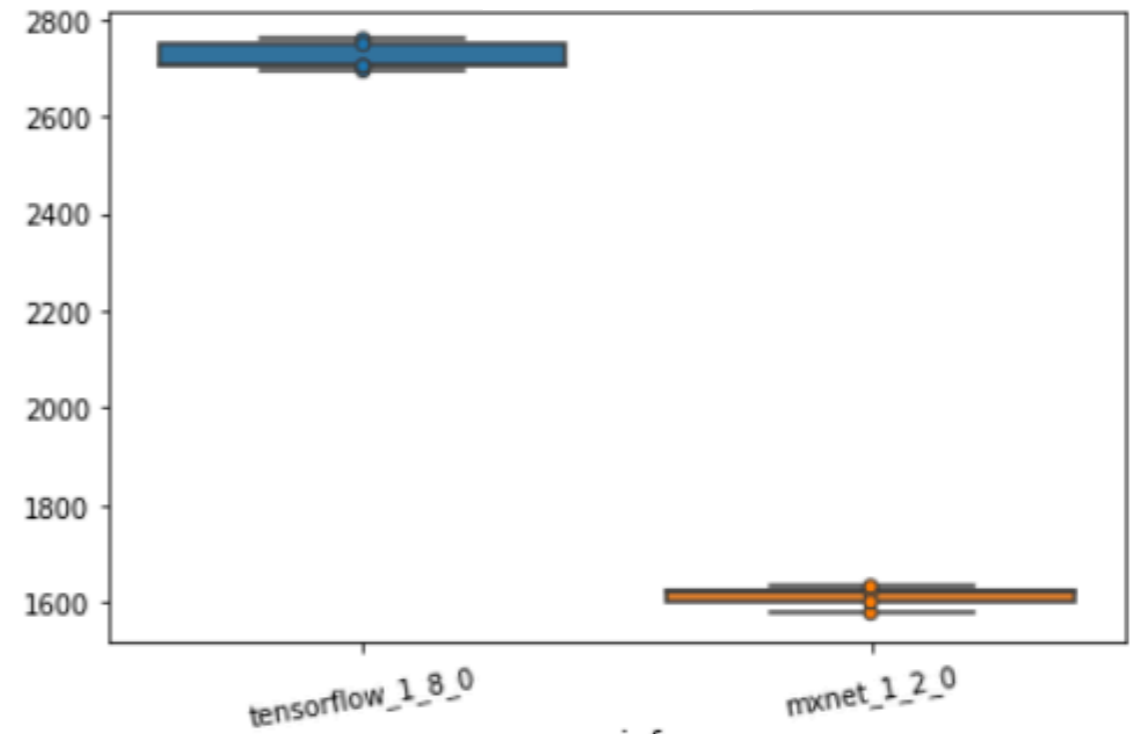
CNN INFER



LSTM TRAIN



LSTM INFER





# Tensorflow - CPU

Line #	Hits	Time	Per Hit	% Time	Line Contents
47					@profile
48					def run_benchmark_train(self, gpus=0):
49					
50	1	2.0	2.0	0.0	x_train, y_train = self.x_train, self.y_train
51					
52	1	5569.0	5569.0	0.4	model = Sequential()
53	1	15978.0	15978.0	1.1	model.add(Dense(512, activation='relu', input_shape=(784,)))
54	1	19014.0	19014.0	1.3	model.add(Dropout(0.2))
55	1	14490.0	14490.0	1.0	model.add(Dense(512, activation='relu'))
56	1	17554.0	17554.0	1.2	model.add(Dropout(0.2))
57	1	14042.0	14042.0	1.0	model.add(Dense(self.num_classes))
58					
59	1	3.0	3.0	0.0	if keras.backend.backend() is "tensorflow" and gpus > 1:
60					model = multi_gpu_model(model, gpus=gpus)
61					
62	1	0.0	0.0	0.0	model.compile(loss='categorical_crossentropy',
63	1	10901.0	10901.0	0.8	optimizer=RMSprop(),
64	1	28131.0	28131.0	1.9	metrics=['accuracy'])
65					
66					# create a distributed trainer for cntk
67	1	2.0	2.0	0.0	if keras.backend.backend() is "cntk" and gpus > 1:
68					start, end = cntk_gpu_mode_config(model, x_train.shape[0])
69					x_train = x_train[start: end]
70					y_train = y_train[start: end]
71					
72	1	3.0	3.0	0.0	time_callback = timehistory.TimeHistory()
73					
74	1	2.0	2.0	0.0	model.fit(x_train, y_train, batch_size=self.batch_size,
75	1	1.0	1.0	0.0	epochs=self.epochs, verbose=1,
76	1	1326038.0	1326038.0	91.3	callbacks=[time_callback])
77					
78	1	5.0	5.0	0.0	self.fit_time = 0
79	2	3.0	1.5	0.0	for i in range(1, self.epochs):
80	1	1.0	1.0	0.0	self.fit_time += time_callback.times[i]
81					
82	1	0.0	0.0	0.0	self.model = model

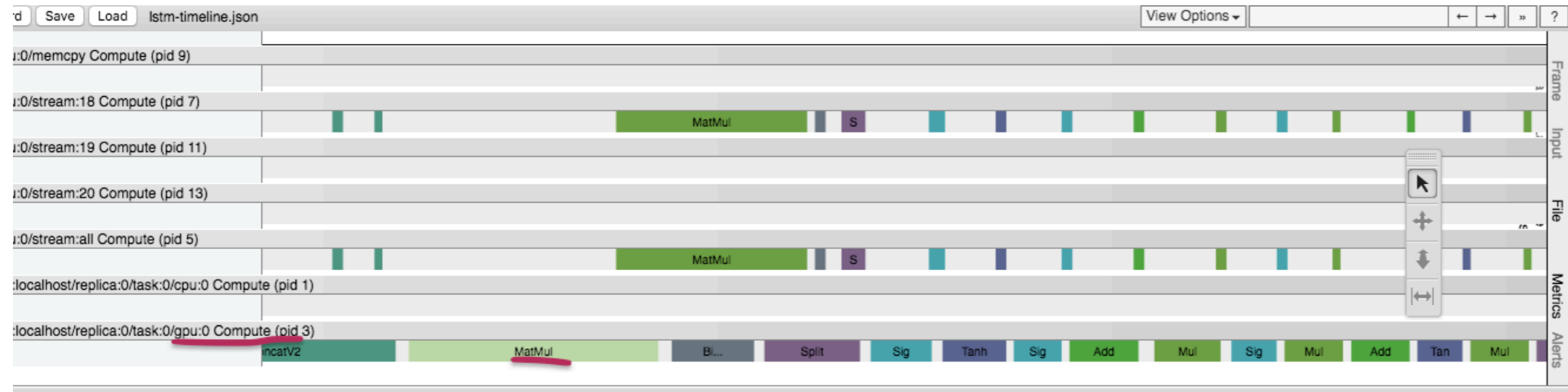
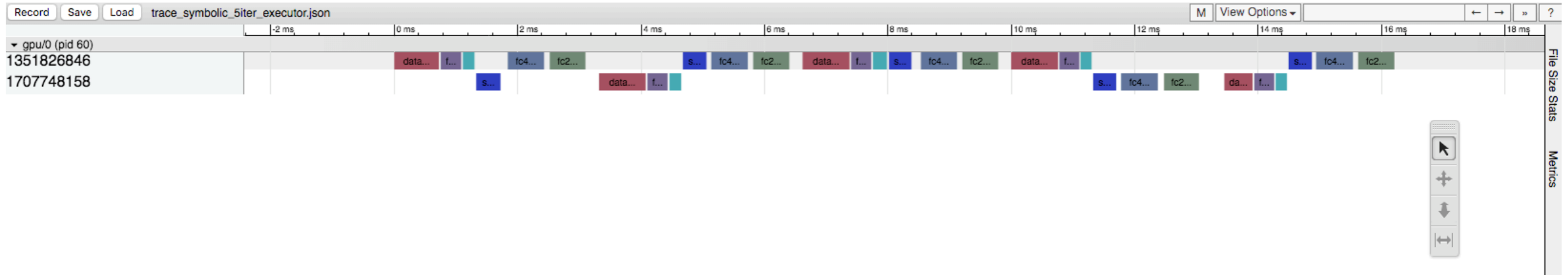
# Tensorflow - GPU

Line #	Hits	Time	Per Hit	% Time	Line Contents
47					@profile
48					def run_benchmark_train(self, gpus=0):
49					
50	1	2.0	2.0	0.0	x_train, y_train = self.x_train, self.y_train
51					
52	1	5986.0	5986.0	0.5	model = Sequential()
53	1	16223.0	16223.0	1.5	model.add(Dense(512, activation='relu', input_shape=(784,)))
54	1	19485.0	19485.0	1.8	model.add(Dropout(0.2))
55	1	14582.0	14582.0	1.3	model.add(Dense(512, activation='relu'))
56	1	17566.0	17566.0	1.6	model.add(Dropout(0.2))
57	1	14073.0	14073.0	1.3	model.add(Dense(self.num_classes))
58					
59	1	3.0	3.0	0.0	if keras.backend.backend() is "tensorflow" and gpus > 1:
60					model = multi_gpu_model(model, gpus=gpus)
61					
62	1	1.0	1.0	0.0	model.compile(loss='categorical_crossentropy',
63	1	11079.0	11079.0	1.0	optimizer=RMSprop(),
64	1	28649.0	28649.0	2.6	metrics=['accuracy'])
65					
66					# create a distributed trainer for cntk
67	1	2.0	2.0	0.0	if keras.backend.backend() is "cntk" and gpus > 1:
68					start, end = cntk_gpu_mode_config(model, x_train.shape[0])
69					x_train = x_train[start: end]
70					y_train = y_train[start: end]
71					
72	1	3.0	3.0	0.0	time_callback = timehistory.TimeHistory()
73					
74	1	1.0	1.0	0.0	model.fit(x_train, y_train, batch_size=self.batch_size,
75	1	0.0	0.0	0.0	epochs=self.epochs, verbose=1,
76	1	972181.0	972181.0	88.4	callbacks=[time_callback])
77					
78	1	5.0	5.0	0.0	self.fit_time = 0
79	2	5.0	2.5	0.0	for i in range(1, self.epochs):
80	1	1.0	1.0	0.0	self.fit_time += time_callback.times[i]
81					
82	1	1.0	1.0	0.0	self.model = model

# mxnet - GPU

```
Line # Hits Time Per Hit % Time Line Contents
=====
47 @profile
48 def run_benchmark_train(self, gpus=0):
49
50 1 2.0 2.0 0.0 x_train, y_train = self.x_train, self.y_train
51
52 1 63347.0 63347.0 16.7 model = Sequential()
53 1 8784.0 8784.0 2.3 model.add(Dense(512, activation='relu', input_shape=(784,)))
54 1 583.0 583.0 0.2 model.add(Dropout(0.2))
55 1 6107.0 6107.0 1.6 model.add(Dense(512, activation='relu'))
56 1 600.0 600.0 0.2 model.add(Dropout(0.2))
57 1 4692.0 4692.0 1.2 model.add(Dense(self.num_classes))
58
59 1 8.0 8.0 0.0 if keras.backend.backend() is "tensorflow" and gpus > 1:
60 model = multi_gpu_model(model, gpus=gpus)
61
62 1 1.0 1.0 0.0 model.compile(loss='categorical_crossentropy',
63 optimizer=RMSprop(),
64 metrics=['accuracy'])
65
66 # create a distributed trainer for cntk
67 1 2.0 2.0 0.0 if keras.backend.backend() is "cntk" and gpus > 1:
68 start, end = cntk_gpu_mode_config(model, x_train.shape[0])
69 x_train = x_train[start: end]
70 y_train = y_train[start: end]
71
72 1 5.0 5.0 0.0 time_callback = timehistory.TimeHistory()
73
74 1 1.0 1.0 0.0 model.fit(x_train, y_train, batch_size=self.batch_size,
75 epochs=self.epochs, verbose=1,
76 callbacks=[time_callback])
77
78 1 5.0 5.0 0.0 self.fit_time = 0
79 2 3.0 1.5 0.0 for i in range(1, self.epochs):
80 1 1.0 1.0 0.0 self.fit_time += time_callback.times[i]
81
82 1 0.0 0.0 0.0 self.model = model
```

# nvprof



**Thank you**