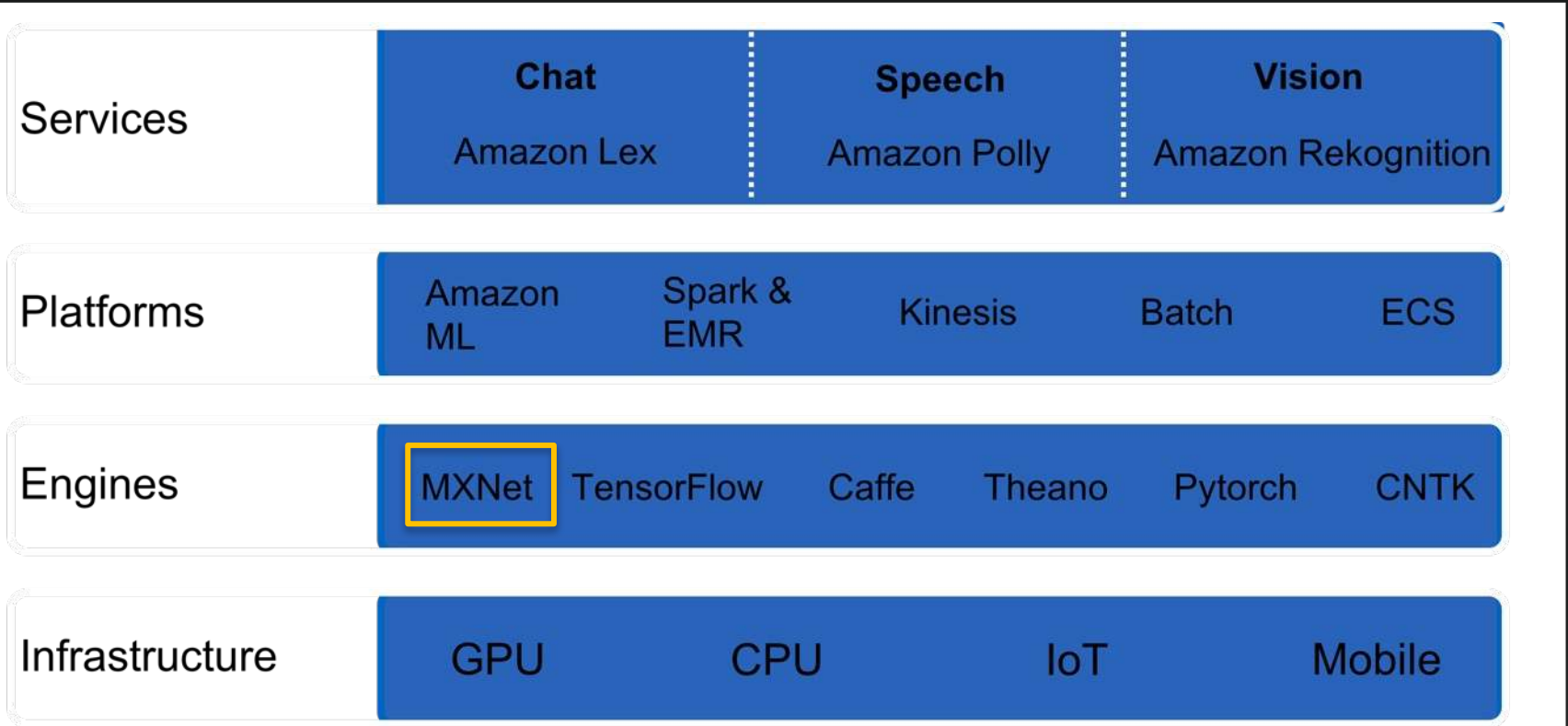
Deep Learning for Developers

Julien Simon <@julsimon>
Principal Evangelist, AI/ML, EMEA

aws

Amazon AI for every developer



The Nvidia V100 GPU is now available on AWS

Launched October 25th

Model	NVIDIA Tesla V100 GPUs	GPU Memory	NVIDIA NVLink	vCPUs	Main Memory	Network Bandwidth	EBS Bandwidth
p3.2xlarge	1	16 GiB	n/a	8	61 GiB	Up to 10 Gbps	1.5 Gbps
p3.8xlarge	4	64 GiB	200 GBps	32	244 GiB	10 Gbps	7 Gbps
p3.16xlarge	8	128 GiB	300 GBps	64	488 GiB	25 Gbps	14 Gbps

```
+-----------------------------------------------------------------------------+
| NVIDIA-SMI 384.81                 Driver Version: 384.81                    |
|-------------------------------+----------------------+----------------------+
| GPU  Name        Persistence-M| Bus-Id        Disp.A | Volatile Uncorr. ECC |
| Fan  Temp  Perf  Pwr:Usage/Cap|         Memory-Usage | GPU-Util  Compute M. |
|===============================+======================+======================|
|   0  Tesla V100-SXM2...  On   | 00000000:00:1B.0 Off |                    0 |
| N/A   45C    P0    35W / 300W |      0MiB / 16152MiB |      0%      Default |
+-------------------------------+----------------------+----------------------+
|   1  Tesla V100-SXM2...  On   | 00000000:00:1C.0 Off |                    0 |
| N/A   40C    P0    34W / 300W |      0MiB / 16152MiB |      0%      Default |
+-------------------------------+----------------------+----------------------+
|   2  Tesla V100-SXM2...  On   | 00000000:00:1D.0 Off |                    0 |
| N/A   41C    P0    38W / 300W |      0MiB / 16152MiB |      0%      Default |
+-------------------------------+----------------------+----------------------+
|   3  Tesla V100-SXM2...  On   | 00000000:00:1E.0 Off |                    0 |
| N/A   43C    P0    39W / 300W |      0MiB / 16152MiB |      0%      Default |
+-------------------------------+----------------------+----------------------+
```

```
INFO:root:Epoch[7] Validation-accuracy=0.991587
INFO:root:Epoch[8] Train-accuracy=0.997513
INFO:root:Epoch[8] Time cost=2.519
INFO:root:Epoch[8] Validation-accuracy=0.991687
INFO:root:Epoch[9] Train-accuracy=0.998114
INFO:root:Epoch[9] Time cos
INFO:root:Epoch[9] Validati
INFO:root:Saved checkpoint
('accuracy', 0.991386217948
```

```
INFO:root:Epoch[7] Train-accuracy=0.996628
INFO:root:Epoch[7] Time cost=1.270
INFO:root:Epoch[7] Validation-accuracy=0.992488
INFO:root:Epoch[8] Train-accuracy=0.997246
INFO:root:Epoch[8] Time cost=1.273
INFO:root:Epoch[8] Validation-accuracy=0.992488
INFO:root:Epoch[9] Train-accuracy=0.997680
INFO:root:Epoch[9] Time cost=1.271
INFO:root:Epoch[9] Validation-accuracy=0.992388
INFO:root:Saved checkpoint to "lenet-0010.params"
('accuracy', 0.9923878205128205)
```

MNIST on 1 GPU:
2x speedup vs p2

https://aws.amazon.com/blogs/aws/new-amazon-ec2-instances-with-up-to-8-nvidia-tesla-v100-gpus-p3/

https://devblogs.nvidia.com/parallelforall/inside-volta/

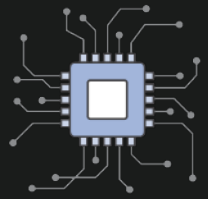
Apache MXNet: Open Source library for Deep Learning

Programmable

Simple syntax, multiple languages

Portable

Highly efficient models for mobile and IoT



High Performance

Near linear scaling across hundreds of GPUs

Most Open

Accepted into the Apache Incubator

Best On AWS

Optimized for Deep Learning on AWS

https://mxnet.io

Last June, tuSimple drove an autonomous truck

for 200 miles from Yuma, AZ to San Diego,

https://www.oreilly.com/ideas/self-driving-trucks-enter-the-fast-lane-using-deep-learning

mx.model.FeedForward

model.fit

Input

Output

mx.sym.SoftmaxOutput

mx.sym.**Activation**(data, act_type="xxxx")

"sigmoid"

"tanh"

"relu"

"softrelu"

mx.sym.**FullyConnected**(data, num_hidden=128)

mx.sym.**Convolution**(data, kernel=(5,5), num_filter=20)

mx.sym.**Pooling**(data, pool_type="max", kernel=(2,2),

stride=(2,2)

lstm.lstm_unroll(num_lstm_layer, seq_len, len, num_hidden, num_embed)

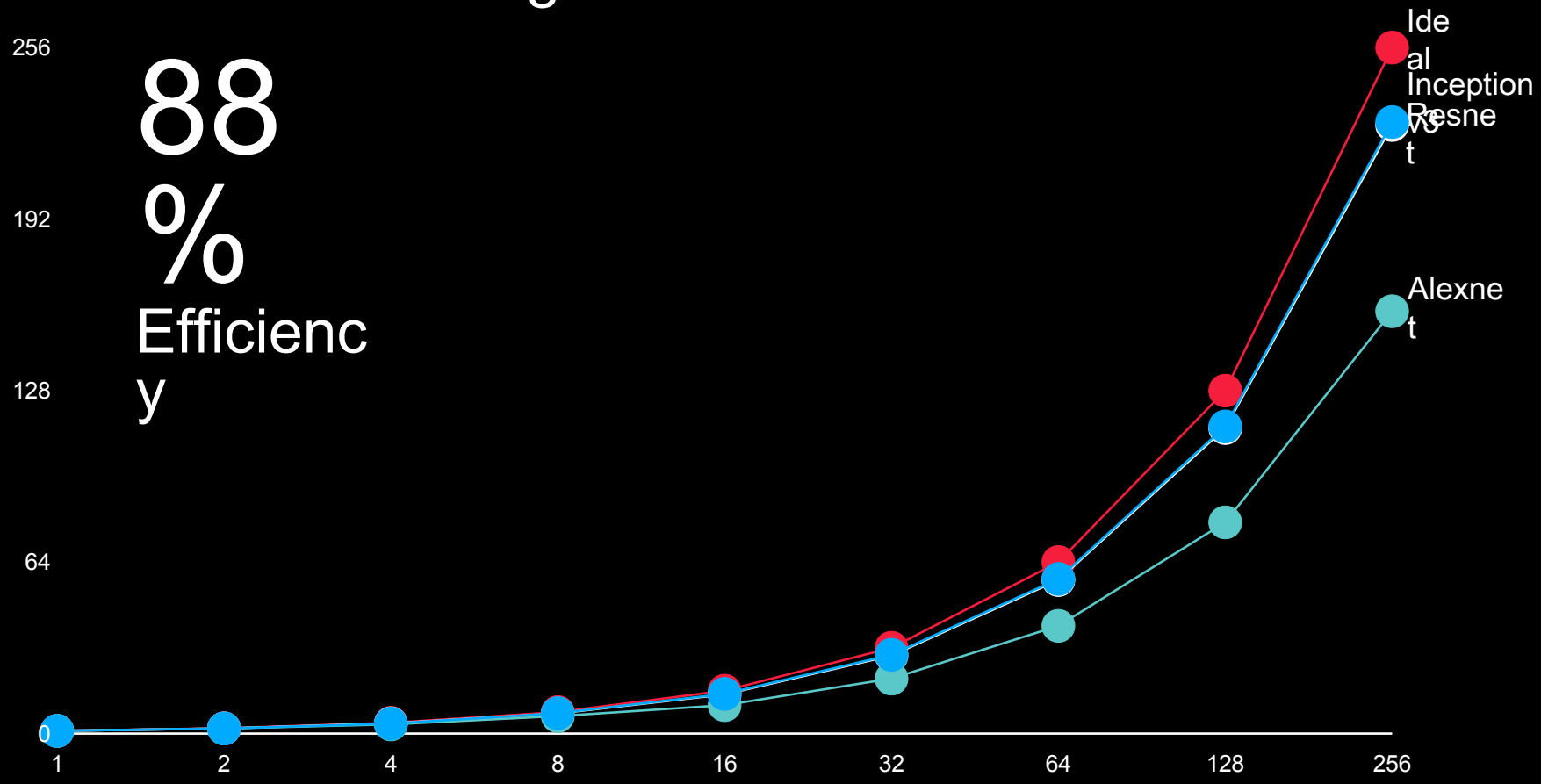
$cos(w, \textbf{queen}) = cos(w, \textbf{king}) - cos(w, \textbf{man}) + cos(w, \textbf{woman})$

mx.symbol.**Embedding**(data, input_dim, output_dim = k)

CPU or GPU: your choice

```
mod = mx.mod.Module(lenet)

mod = mx.mod.Module(lenet, context=mx.gpu(0))

mod = mx.mod.Module(lenet,
context=(mx.gpu(7), mx.gpu(8), mx.gpu(9)))
```

Multi-GPU Scaling With MXNet



AWS Deep Learning AMI

- Deep Learning Frameworks – Popular Deep Learning Frameworks (MXNet, Caffe, Tensorflow, Theano, Torch, etc.) all prebuilt and pre-installed
- GPU components – Nvidia drivers, cuDNN, CUDA 8 & 9
- AWS Integration – Packages and configurations that provide tight integration with Amazon Web Services
- Amazon Linux & Ubuntu

https://aws.amazon.com/blogs/ai/announcing-new-aws-deep-learning-ami-for-amazon-ec2-p3-instances/

Apache MXNet demos

1. Image classification: using pre-trained models
   Imagenet, multiple CNNs, MXNet
2. Image classification: fine-tuning a pre-trained model
   CIFAR-10, ResNet-50, Keras + MXNet
3. Image classification: learning from scratch
   MNIST, MLP & LeNet, MXNet
4. Machine Translation: translating German to English
   News, LSTM, Sockeye + MXNet

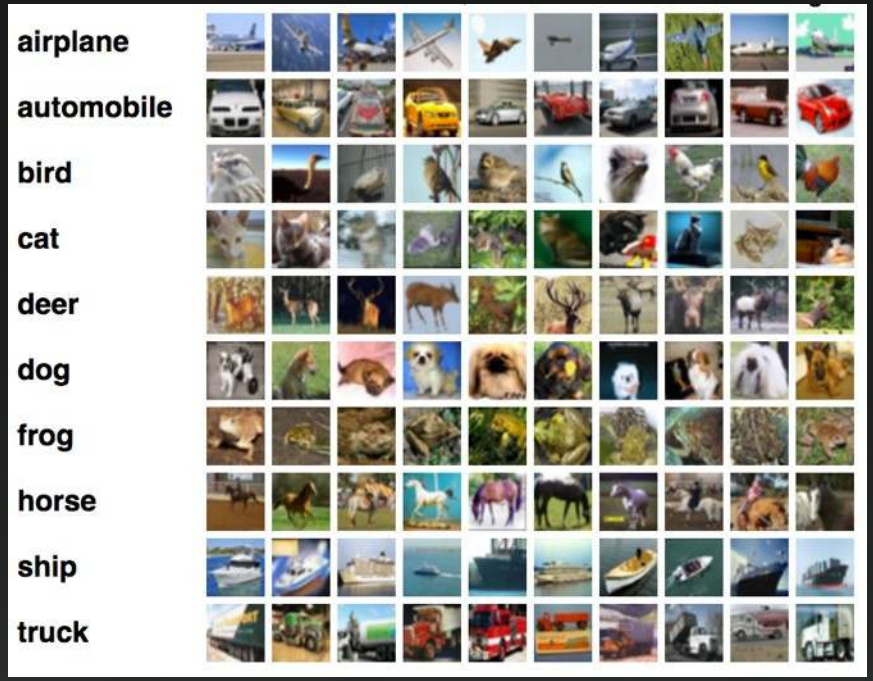
Demo #1 – Image classification: using a pre-trained model

```
*** VGG16
[(0.46811387, 'n04296562 stage'), (0.24333163,
'n03272010 electric guitar'), (0.045918692, 'n02231487
walking stick, walkingstick, stick insect'),
(0.03316205, 'n04286575 spotlight, spot'),
(0.021694135, 'n03691459 loudspeaker, speaker, speaker
unit, loudspeaker system, speaker system')]
*** ResNet-152
[(0.8726753, 'n04296562 stage'), (0.046159592,
'n03272010 electric guitar'), (0.041658506, 'n03759954
microphone, mike'), (0.018624334, 'n04286575 spotlight,
spot'), (0.0058045341, 'n02676566 acoustic guitar')]
*** Inception v3
[(0.44991142, 'n04296562 stage'), (0.43065304,
'n03272010 electric guitar'), (0.067580454, 'n04456115
torch'), (0.012423956, 'n02676566 acoustic guitar'),
(0.0093934005, 'n03250847 drumstick')]
```

https://medium.com/@julsimon/an-introduction-to-the-mxnet-api-part-5-9e78534096db

Demo #2 – Image classification: fine-tuning a model

- CIFAR-10 data set
  - 60,000 images in 10 classes
  - 32x32 color images
- Initial training
  - Resnet-50 CNN
  - 200 epochs
  - 82.12% validation
- Cars vs. horses
  - 88.8% validation accuracy



https://medium.com/@julsimon/keras-shoot-out-part-3-fine-tuning-7d1548c51a41

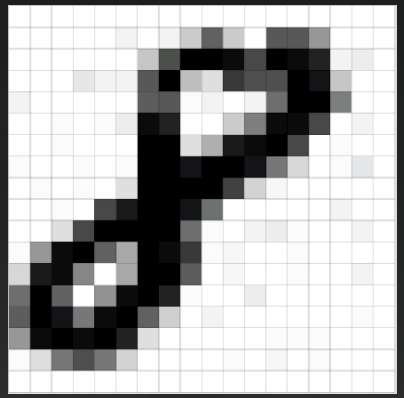
Demo #2 – Image classification: fine-tuning a model

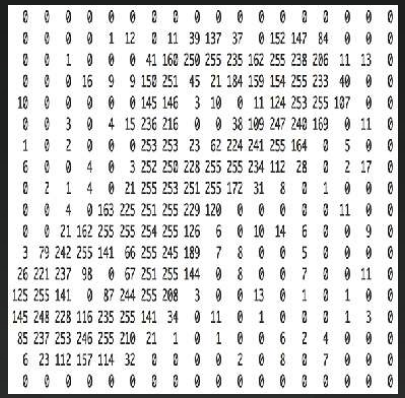
- Freezing all layers but the last one
- Fine-tuning on « cars vs. horses » for 10 epochs
- 2 minutes on 1 GPU
- 98.8% validation accuracy

```
Epoch 10/10
10000/10000 [==============================] - 12s
loss: 1.6989 - acc: 0.9994 - val_loss: 1.7490 - val_acc: 0.9880

2000/2000 [==============================] - 2s
[1.7490020694732666, 0.98799999999999999]
```

Demo #3 – Image classification: learning from scratch

- MNIST data set
- 70,000 hand-written digits
- 28x28 grayscale images

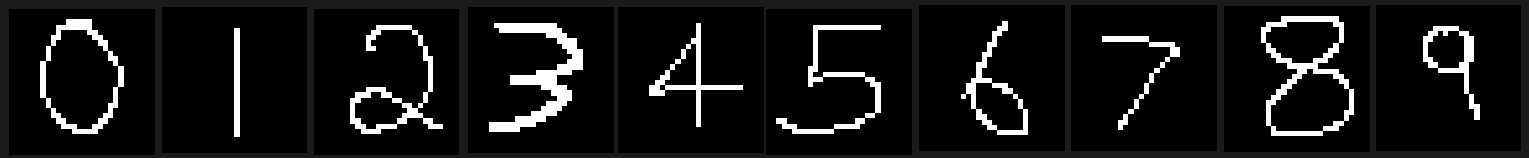




https://medium.com/@julsimon/training-mxnet-part-1-mnist-6f0dc4210c62

Multi-Layer Perceptron vs. Handmade-Digits-From-Hell™

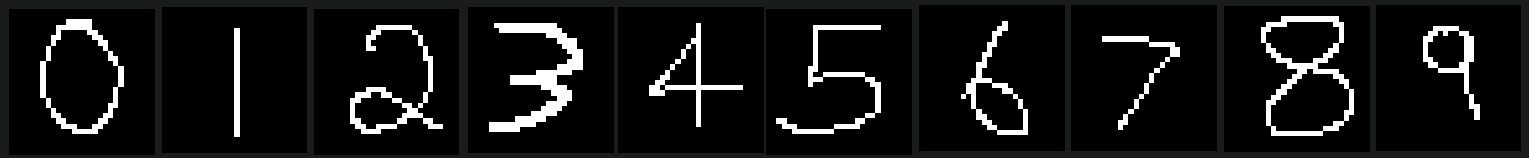
784/128/64/10, Relu, AdaGrad, 100 epochs → 97.51% validation accuracy



```
[[ 0.839  0.034  0.039  0.009  0.     0.008  0.066  0.002  0.     0.004]]
[[ 0.     0.988  0.001  0.003  0.001  0.001  0.002  0.003  0.001  0.002]]
[[ 0.006  0.01   0.95   0.029  0.     0.001  0.004  0.     0.     0.]]
[[ 0.     0.     0.     1.     0.     0.     0.     0.     0.     0.]]
[[ 0.     0.001  0.005  0.001  0.982  0.001  0.     0.007  0.     0.002]]
[[ 0.001  0.001  0.     0.078  0.     0.911  0.01   0.     0.     0.]]
[[ 0.003  0.     0.019  0.     0.005  0.004  0.863  0.     0.105  0.001]]
[[ 0.001  0.008  0.098  0.033  0.     0.     0.     0.852  0.004  0.004]]
[[ 0.001  0.     0.006  0.     0.     0.001  0.002  0.     0.991  0.]]
[[ 0.002  0.158  0.007  0.117  0.082  0.001  0.     0.239  0.17   0.224]]
```

LeNet CNN vs. Handmade-Digits-From-Hell™

ReLu instead of tanh, 10 epochs, AdaGrad → 99.20% validation accuracy



```
[[ 1.      0.      0.      0.      0.      0.      0.      0.      0.      0.]]
[[ 0.      1.      0.      0.      0.      0.      0.      0.      0.      0.]]
[[ 0.      0.      1.      0.      0.      0.      0.      0.      0.      0.]]
[[ 0.      0.      0.      1.      0.      0.      0.      0.      0.      0.]]
[[ 0.      0.      0.001   0.      0.998   0.      0.      0.001   0.      0.]]
[[ 0.      0.      0.      0.      0.      1.      0.      0.      0.      0.]]
[[ 0.      0.      0.      0.      0.      0.      1.      0.      0.      0.]]
[[ 0.      0.      0.      0.001   0.      0.      0.      0.999   0.      0.]]
[[ 0.      0.      0.006   0.      0.      0.      0.      0.      0.994   0.]]
[[ 0.      0.      0.      0.001   0.001   0.      0.      0.001   0.001   0.996]]
```

aws

Demo #4 – Machine Translation: German to English

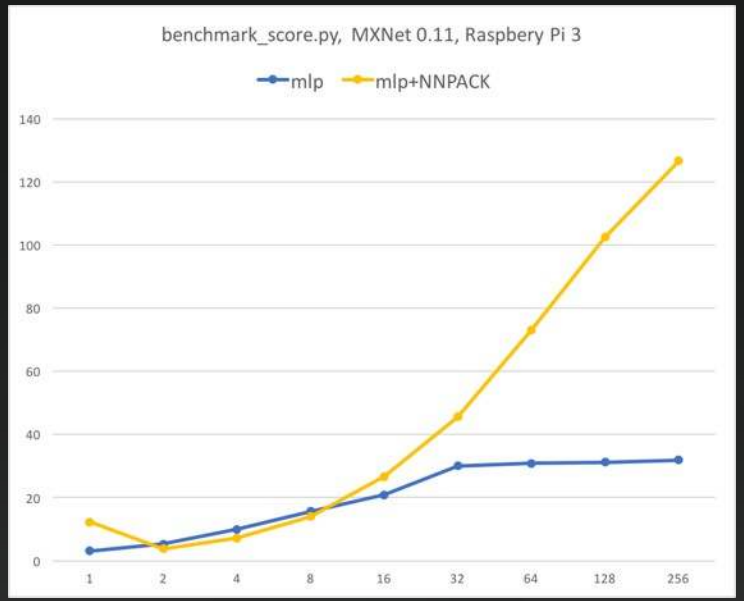
- AWS Open Source project https://github.com/awslabs/sockeye
- Sequence-to-sequence models with Apache MXNet
- 5.8M sentences (news headlines), 5 hours of training on 8 GPUs (p2)

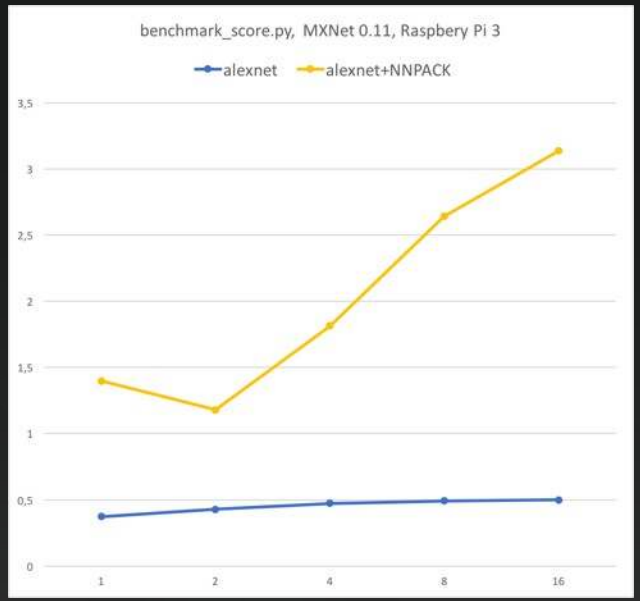
```
./translate.sh "Chopin zählt zu den bedeutendsten Persönlichkeiten der
Musikgeschichte Polens ."

Chopin is one of the most important personalities of Poland's history

./translate.sh "Hotelbetreiber müssen künftig nur den Rundfunkbeitrag bezahlen,
wenn ihre Zimmer auch eine Empfangsmöglichkeit bieten ."

in the future , hotel operators must pay only the broadcasting fee if their rooms
also offer a reception facility .
```

https://aws.amazon.com/blogs/ai/train-neural-machine-translation-models-with-sockeye/

Speeding up inference on CPU

- Intel MKL https://software.intel.com/en-us/mkl
- NNPACK https://github.com/Maratyszcza/NNPACK





https://medium.com/@julsimon/speeding-up-apache-mxnet-with-the-nnpack-library-7427f367490f
https://medium.com/@julsimon/speeding-up-apache-mxnet-with-the-nnpack-library-raspberry-pi-edition-e444b446a180

Shrinking models

- Complex neural networks are too large for resource-constrained environments
- MXNet supports Mixed Precision Training
  - Use float16 instead of float32
  - Almost 2x reduction in memory consumption, no loss of accuracy
  - https://devblogs.nvidia.com/parallelforall/mixed-precision-training-deep-neural-networks/
  - http://docs.nvidia.com/deeplearning/sdk/mixed-precision-training/index.html#mxnet
- BMXNet: Binary Neural Network Implementation
  - Use binary values
  - 20x to 30x reduction in model size, with limited loss

om/hpi-xnor/BMXNet

Model	Baseline	Mixed Precision
AlexNet	56.77%	56.93%
VGG-D	65.40%	65.43%
GoogleNet	68.33%	68.43%
Inception v1	70.03%	70.02%
Resnet50	73.61%	73.75%

	Architecture	Test Accuracy (Binary/Full Precision)	Model Size (Binary/Full Precision)
MNIST	Lenet	0.97/0.99	206kB/4.6MB
CIFAR-10	ResNet-18	0.86/0.90	1.5MB/44.7MB

Gluon: Deep Learning gets even easier

https://github.com/gluon-api/

- Announced October 11th
- Available now in MXNet, soon in Microsoft Cognitive Toolkit

- Developer-friendly high-level API
- No compromise on performance
- Networks can be modified during training
- Extensive model zoo

https://aws.amazon.com/blogs/aws/introducing-gluon-a-new-library-for-machine-learning-from-aws-and-microsoft/
https://mxnet.incubator.apache.org/versions/master/api/python/gluon/model_zoo.html

Anything you dream is fiction, and anything you accomplish is science, the whole history of mankind is nothing but science fiction.

Ray Bradbury

Resources

https://aws.amazon.com/ai/
https://aws.amazon.com/blogs/ai/
https://mxnet.io
https://github.com/gluon-api/
https://github.com/awslabs/sockeye

https://reinvent.awsevents.com/ watch this space ;)

https://medium.com/@julsimon/

Thank you!

https://aws.amazon.com/evangelists/julien-simon

@julsimon