CODE - Jaredlander.com/content/2017/06/Feedforward.html, and a few others

Lecturer – Dan Romuald Mbanga, of AWS.

Feedback loop: more data -> better analytics -> better products -> more users -> more data.

More data: click stream, user activity, generated content, purchases clicks likes sensor ata

Analytics: object storage, streaming analytics, BI, big data methods

Products: AI

More users: pinpoint your users, understand them and what they want, improve retention and attraction

AI def Mbanga uses: it’s an umbrella term. Hardware + learning algos broadly, more narrowly ML that fits fct to data, more narrowly DL that fits a network structure to the data

Interaction b/w real world and computer is the goal. E.g. being able to predict word user intends regardless of what exactly they said.

Fast computing, advanced learning algos, and ubiquitous data make this a good time for AI.

Uses of AI at Amazon:

Discovery and search – recommender systems

Fulfillment and logistics, e.g. computer vision for telling pickers which items are where

Enhancing existing products (cf Fire phone buying features)

Define new categories of products (cf Alexa)

Goal to bring ML to all customers

First Amazon recommender system was in 1998.

Today, can do stuff like recognize actors in movie scenes in real time.

Amazon AI platforms:

Services – Rekognition for vision, Polly for speec, Lex for Chat, more planned for this year

Platforms for users to train products w/ their own data: Amazon ML, Spark & EMR, Kinesis, Batch, ECS, more in 2017.

Engines: MXnet, TensorFlow, Caffe, Theano, Pytorch, CNTK

Why do they still use all, rather than standardizing on one?

Infrastructure: GPU, CPU, IoT, mobile

Deep learning has potential for solving some of the hardest problems in comp sci – learning (save us from manual annotation! Save us from manual updates!), language (big fan of neural machine translation), perception, problem solving, reasoning

Found that can basically port activations from one language into another. Interesting!

AWS deep learning AMI – “one click GPU deep learning”

Up to 40k CUDA cores. Tensoflow and MXNet are preferred interfaces, but supports Caffe and so on. Has notebooks and examples, scales for training.

# Mxnet

Advantages of MXnet: flexible (mixed programming API, supports lots of langs), portable (runs everywhere), high performance (near linear scaling - )

Backend is C++, code is compiled from whatever you write it in and runs as it

Have it working on mobile and on drones, and in browsers w/ datascrape

Tensors: blal blah vector but more dimensions

Steps: tensor, neural net, train, optimize, scale, input data

## Tensors

mxnet.nd.array is how you encode your data into tensors.

## Net

Use mxnet.symbol to set up the model.

“Computational graph” – describe the model you want to build, don’t worry about what’s behind the scenes

## Train

Use mxnet.module for your error fct or optimization fct. Also include regularization term here.

I need to check if you just tell it L1 or what

## Optimize

Mxnet.initializer, mxnet.optimizer, mxnet.lr\_scheduler for optimization

Implement momentum, RMSprop, etc. here

Initializer is how you initialize your weights! (Check how it likes you to tell it about fan-in)

Let you set hyperparameters you want. Gives an interesting viz comparing diff methods of improving on simple gradient descent

## Scale

Let’s say you’ve got 4 GPUs. You want to partition your data into 4 using a key-value store.

On a single machine: read data partitions, pull the parameters, compute the gradient, push the gradient back to the key-value store, update the weights.

On a cluster w/ same number of GPUs, no change in the code! You don’t have to write new parallelism code for each change in what it’s running on.

## Input data

mxnet.io method lets you ingest data

mxnet.image

Check if there’s a premade DICOM method? Nice if it has a parameter for DICOMs

Kinesis: esp for online learning, need to think a/b how you want to bring up your data

Upload, direct connect, Kineses (streams, firehose), IoT, database migration

AWS Snowball is hardware for this. Sounds like this is a sneakernet thing

After you get it in, collect and organize: Amazon AI, Aurora, RDSes like MySQL or SQL server

Amazon Glue is a thing for making end to end pipelines

Mechanical Turk is an annotation option

## AWS

EC2 lets you do virtual machines. Have deep learning VMs here. Also can have custom environments like the AMI Mbanga prefers to use.

Mentions a dedicated package for GANs he has

# The code

NYC Planning dataset, split into train validate and test

Jaredlander.com/content/2017/06/Feedforward.html, and a few others

Use vtreat for some automated feature engineering. Can use this to handle imbalanced data. This is useful prproc.

Can make custom loss function, which do in the feedforward example. It’s a log-loss error function.

mx..symbol.Variable(‘data’) %>% mx.symbol.Dropout(p=0.2) %>%

Recommend using pipe - I presume you’d need to nest things in reverse order if you didn’t use pipe

.x.symbol.Dropout for dropout reg

Symbol.FullyConnected is full conn

Symbol.BatchNorm for normalizing the units

Symbol.Activation(act\_type = ‘relu’) is a RELU

Architecture he demonstrates does full conns, normalization, RELU, then dropout. Couple stacks of that, then full conn, then output

mx.symbol.LogisticRegressionOutput(name=’output’)

Note that he’s putting a name argument on every layer. This is good practice, b/c lets them be the arguments in an object that is a less painful way to look at what you’ve built

Assigned all of the above to the name net. Then shows

arguments(net)

Includes the weight and bias for each layer and for the output; the data; and the output labels

Graph.viz(net) #check that this is correct

Makes a flowchart of your net! Very helpful.

mx.set.seed fct randomizer for reproducible results

## Sys.setenv is for CPU vs GPU. Check on how to use this!

ctix is “context” – mx.cpu() vs mx.gpu(). Need to specify key values for what machines you’re running it on.

Eval.metric

Num.round is number of epochs

Learning.rate

Array.layout is how data is stored.

I should def look at this tutorial’s code.