Deep Learning at the Edge

Julien Simon, Principal Evangelist, AI & Machine Learning

@julsimon

Deep Learning at the Edge

1. Flexible experimentation in the Cloud.

2. Scalable training in the Cloud.

3. Good prediction performance at the Edge.

4. Simple deployment of code and model

Flexible experimentation in the Cloud

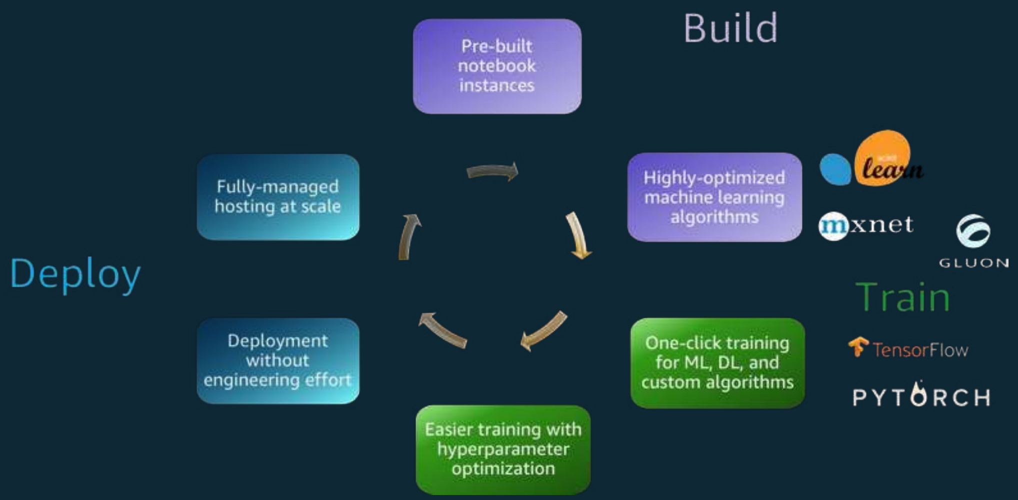
- Apache MXNet: Python, R, Perl, Matlab, Scala, C++.

- Gluon
  - Imperative programming aka ‘define-by-run’.
  - Inspect, debug and modify models during training.

- Extensive model zoo
  - Pre-trained computer vision models.
  - MobileNet, SqueezeNet for resource-constrained devices.

Scalable training in the Cloud

Amazon SageMaker



Build

Pre-built notebook instances

Highly-optimized machine learning algorithms

scikit learn

mxnet

GLUON

Train

One-click training for ML, DL, and custom algorithms

TensorFlow

PYTORCH

Easier training with hyperparameter optimization

Deployment without engineering effort

Fully-managed hosting at scale

Deploy

AWS Deep Learning AMI

TensorFlow

K

Microsoft Cognitive Toolkit

mxnet

GLUON

scikit learn machine learning in Python

Caffe2

PYTORCH

Amazon EC2

c5

p3

aws

Good prediction performance at the Edge

- MXNet is written in C++.
- Gluon networks can be ‘hybridized’ for additional speed.
- Two libraries boost performance on CPU-only devices
    - Fast implementation of math primitives
    - Hardware-specific instructions, e.g. Intel AVX or ARM NEON
    - Intel Math Kernel Library https://software.intel.com/en-us/mkl
    - NNPACK https://github.com/Maratyszcza/NNPACK
- Mixed precision training
    - Use float16 instead of float32 for weights and activations
    - Almost 2x reduction in model size, no loss of accuracy, faster inference
    - https://devblogs.nvidia.com/parallelforall/mixed-precision-training-deep-neural-networks/

Simple deployment of code and model

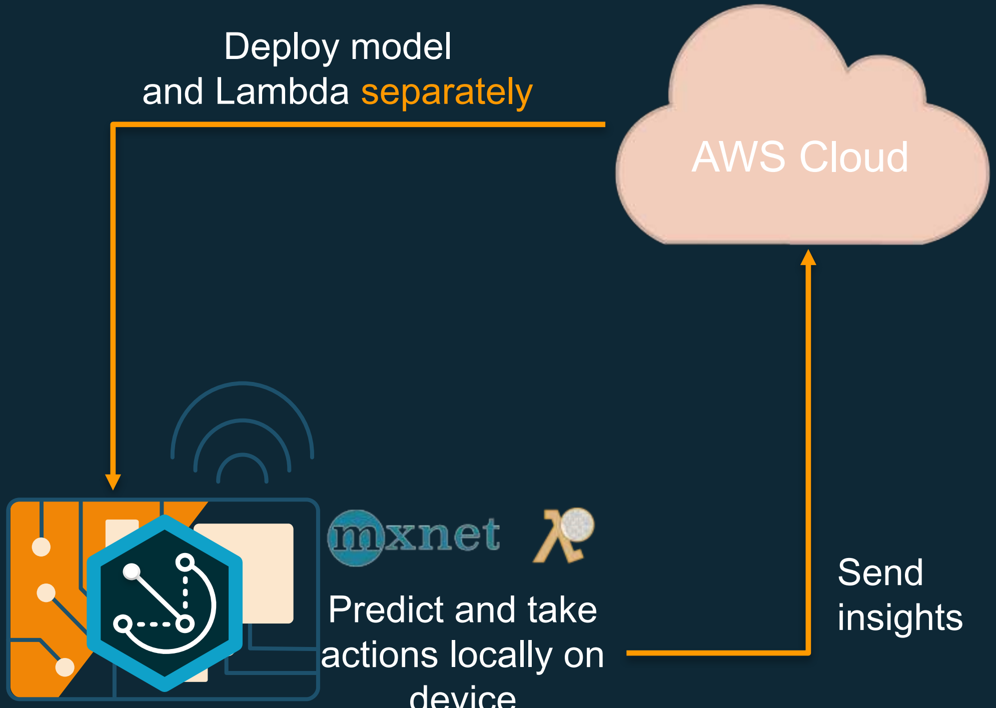
- Train a model in SageMaker (or bring your own).
- Write a Lambda function performing prediction.
- Add both as resources in your Greengrass group.
- Let Greengrass handle deployment and updates.

Best when
You want the same programming model in the Cloud and at the Edge.
Code and models need to be updated, even if network connectivity is infrequent or unreliable.
One device in the group should be able to perform prediction on behalf on other devices.

Requirements
Devices are powerful enough to run Greengrass (XXX HW requirements)
Devices are provisioned in AWS IoT (certificate, keys).

ML Inference using AWS Greengrass

PREVIEW AVAILABLE



AWS DeepLens

Intel Atom CPU

Gen9 graphics

Ubuntu 16.04 LTS

100 GFLOPS performance

Dual band Wi-Fi

8 GB RAM

16 GB Storage (eMMC)

32 GB SD card

4 MP camera with MJPEG

H.264 encoding at 1080p resolution

2 USB ports

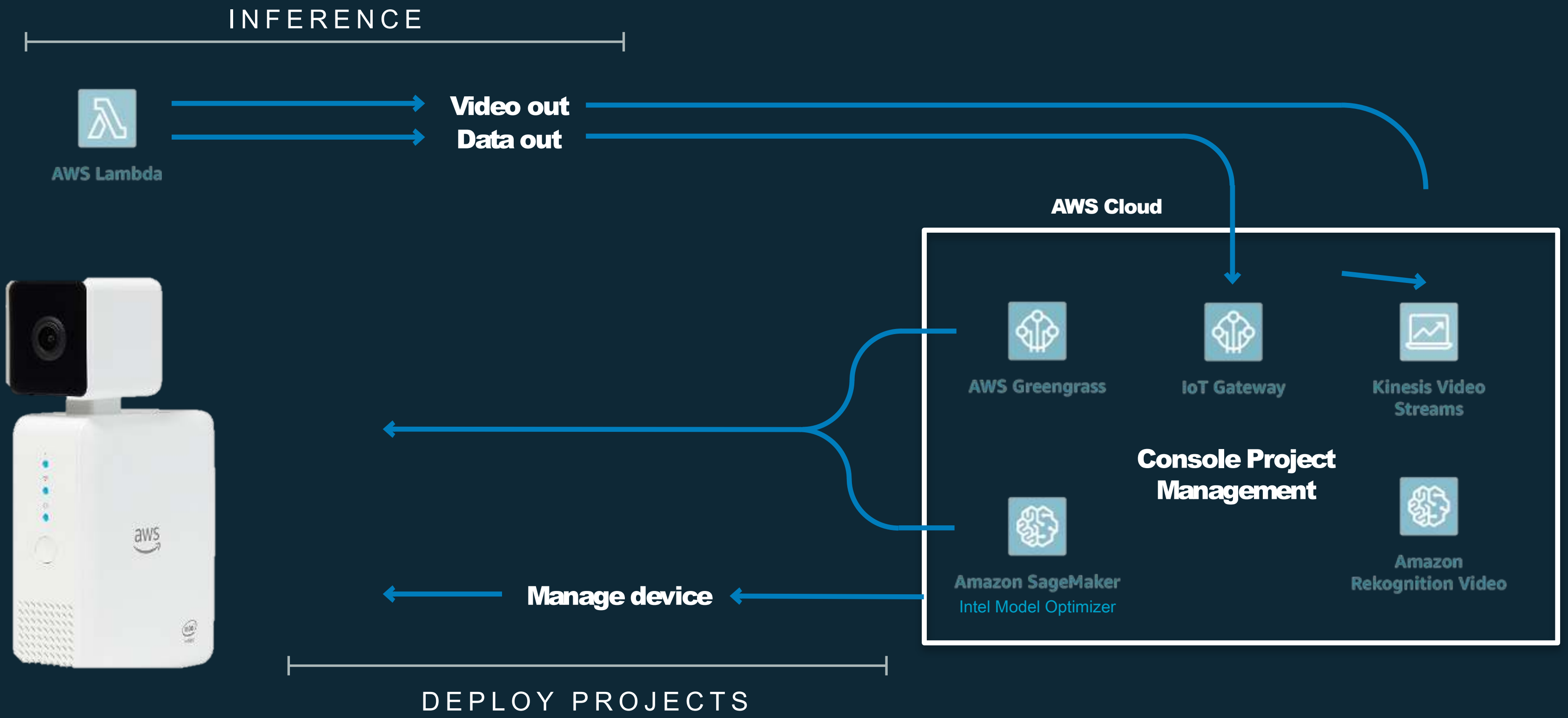
Micro HDMI

Audio out

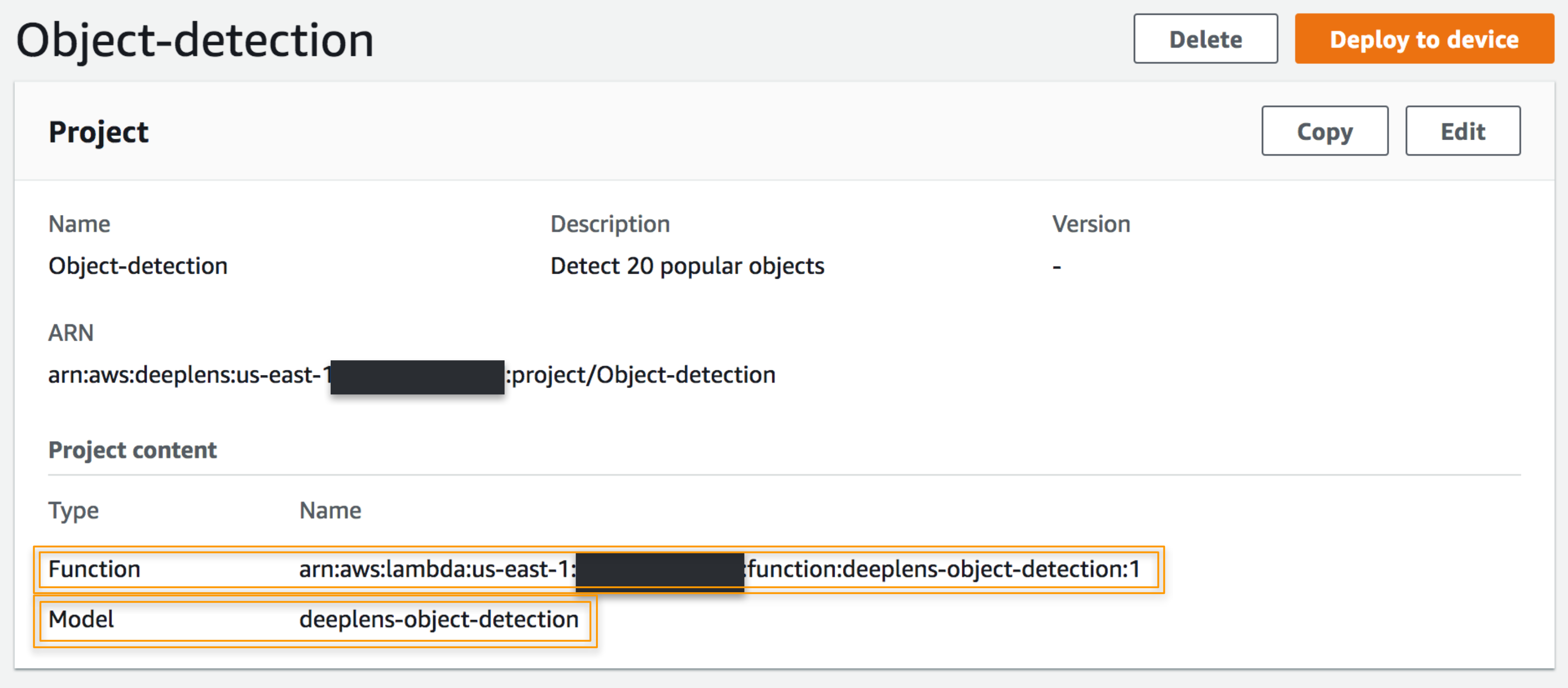
AWS Greengrass preconfigured

Intel clDNN for Apache MXNet

AWS DeepLens Architecture



AWS DeepLens



Object-detection

Project

Name
Object-detection

Description
Detect 20 popular objects

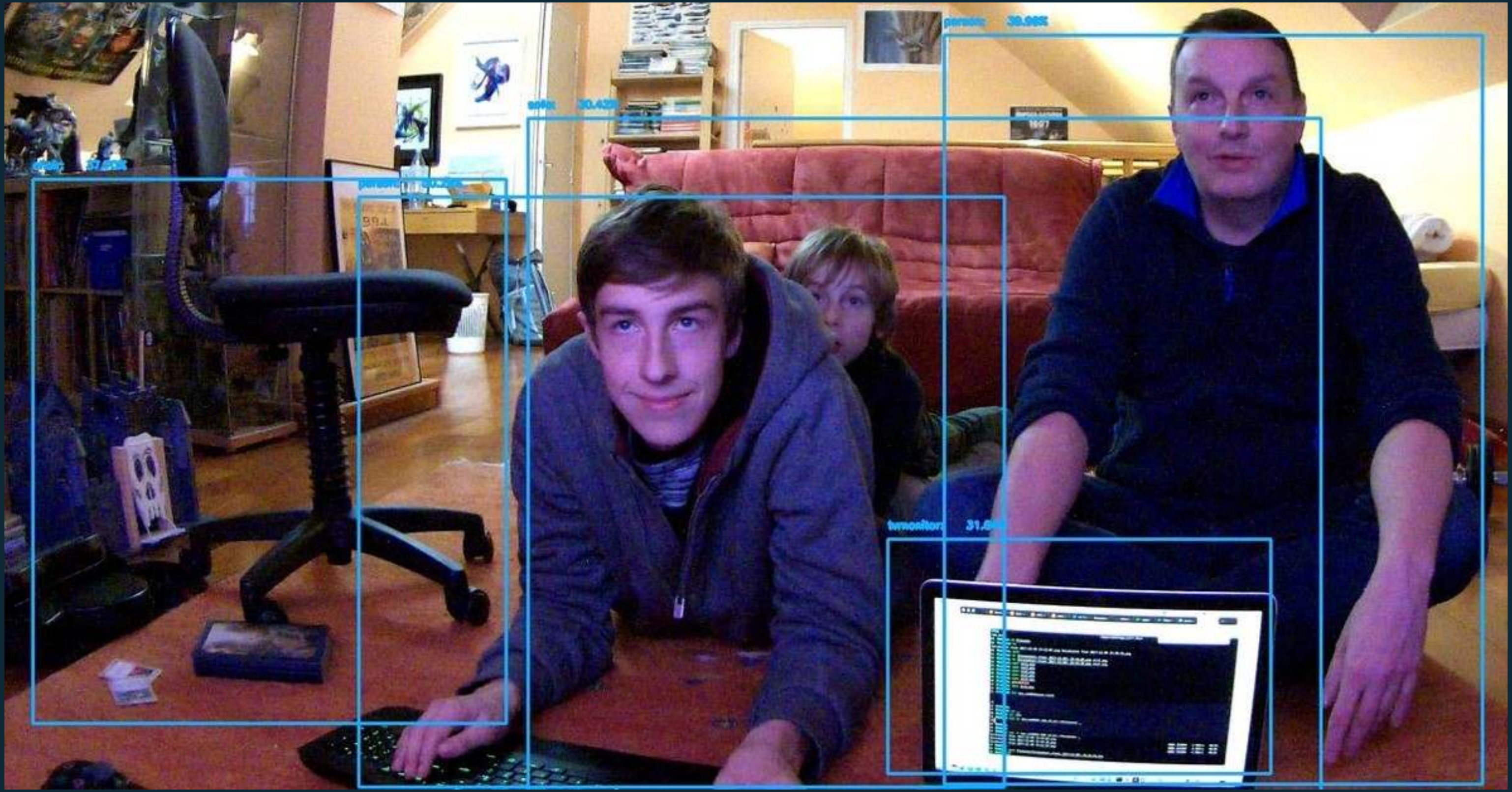
Version
-

ARN
arn:aws:deeplens:us-east-1[redacted]:project/Object-detection

Project content

Type	Name
Function	arn:aws:lambda:us-east-1:[redacted]:function:deeplens-object-detection:1
Model	deeplens-object-detection

Object detection with AWS DeepLens



Resources

Resources

https://mxnet.incubator.apache.org

http://gluon.mxnet.io

https://aws.amazon.com/sagemaker (free tier available)

An overview of Amazon SageMaker: https://www.youtube.com/watch?v=ym7NEYEx9x4

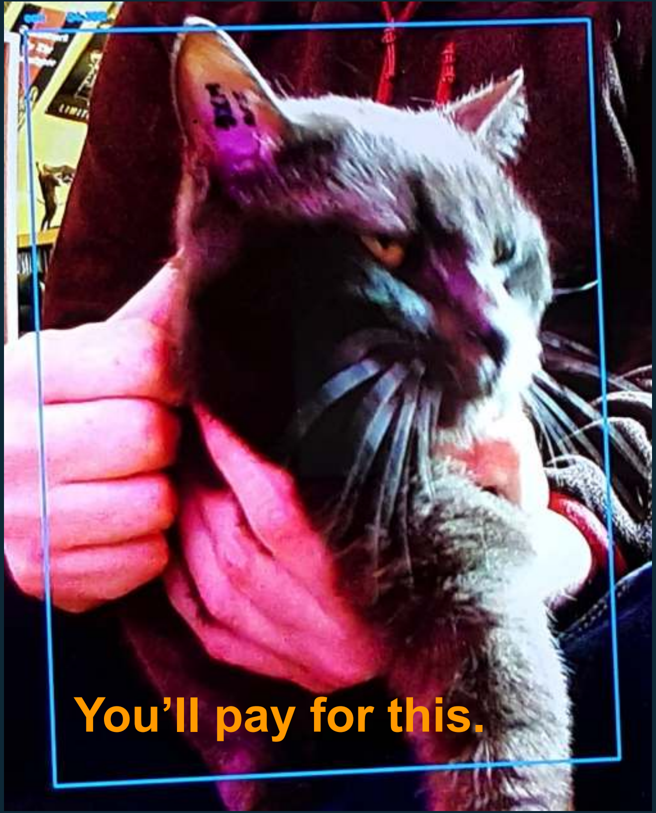
https://github.com/awslabs/amazon-sagemaker-examples

https://aws.amazon.com/greengrass (free tier available)

https://aws.amazon.com/deeplens

https://github.com/intel/clDNN

https://medium.com/@julsimon



Thank you!

Julien Simon, Principal Evangelist, AI & Machine Learning

@julsimon