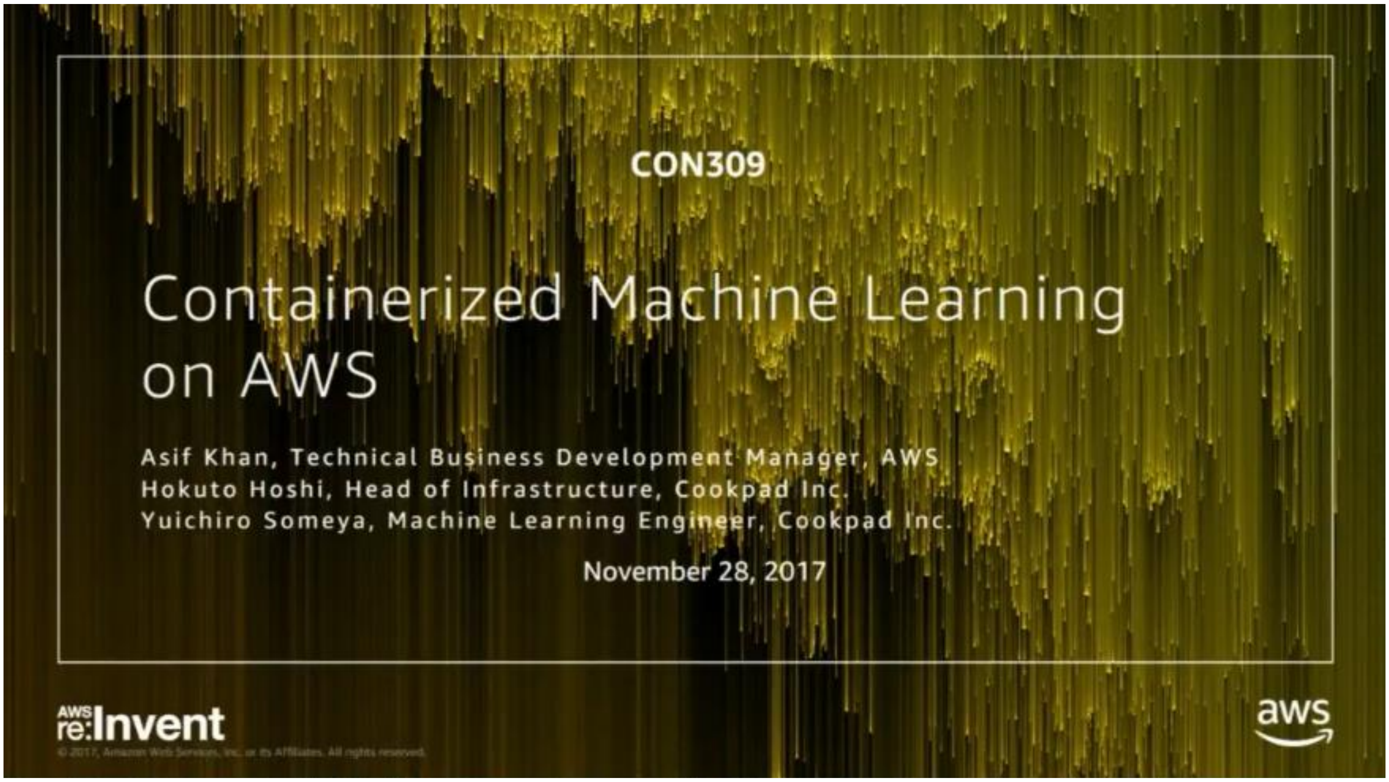


CON309

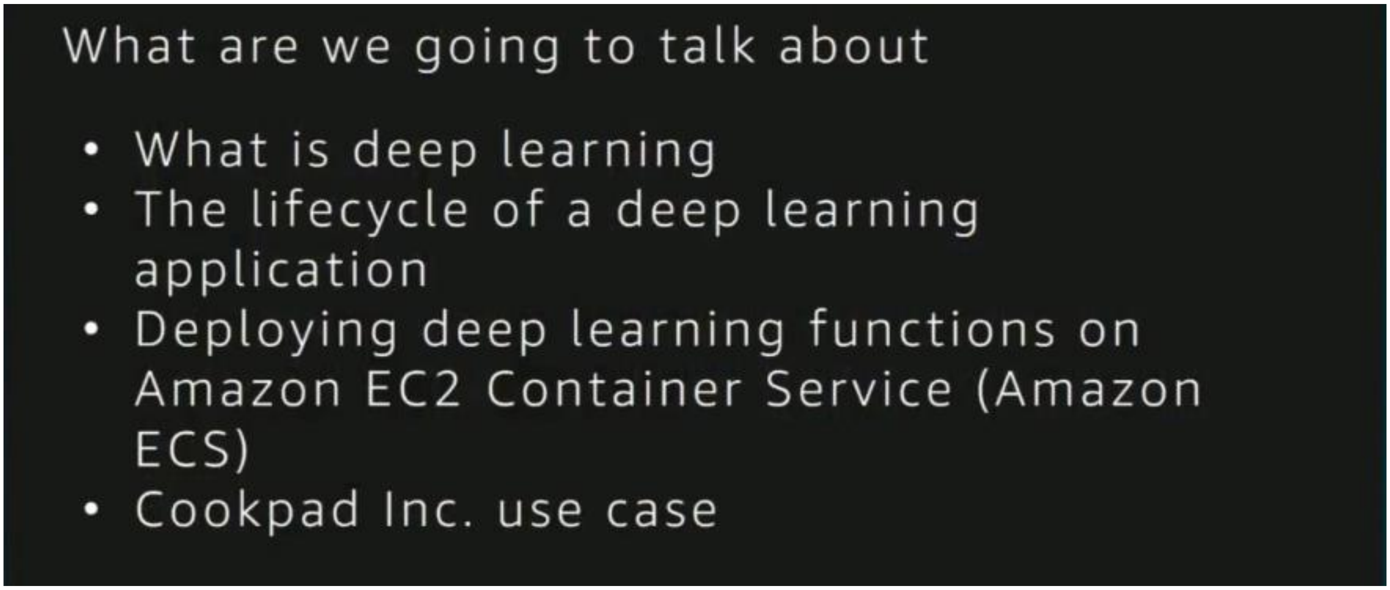
Containerized Machine Learning on AWS

Asif Khan, Technical Business Development Manager, AWS
Hokuto Hoshi, Head of Infrastructure, Cookpad Inc.
Yuichiro Someya, Machine Learning Engineer, Cookpad Inc.

November 28, 2017

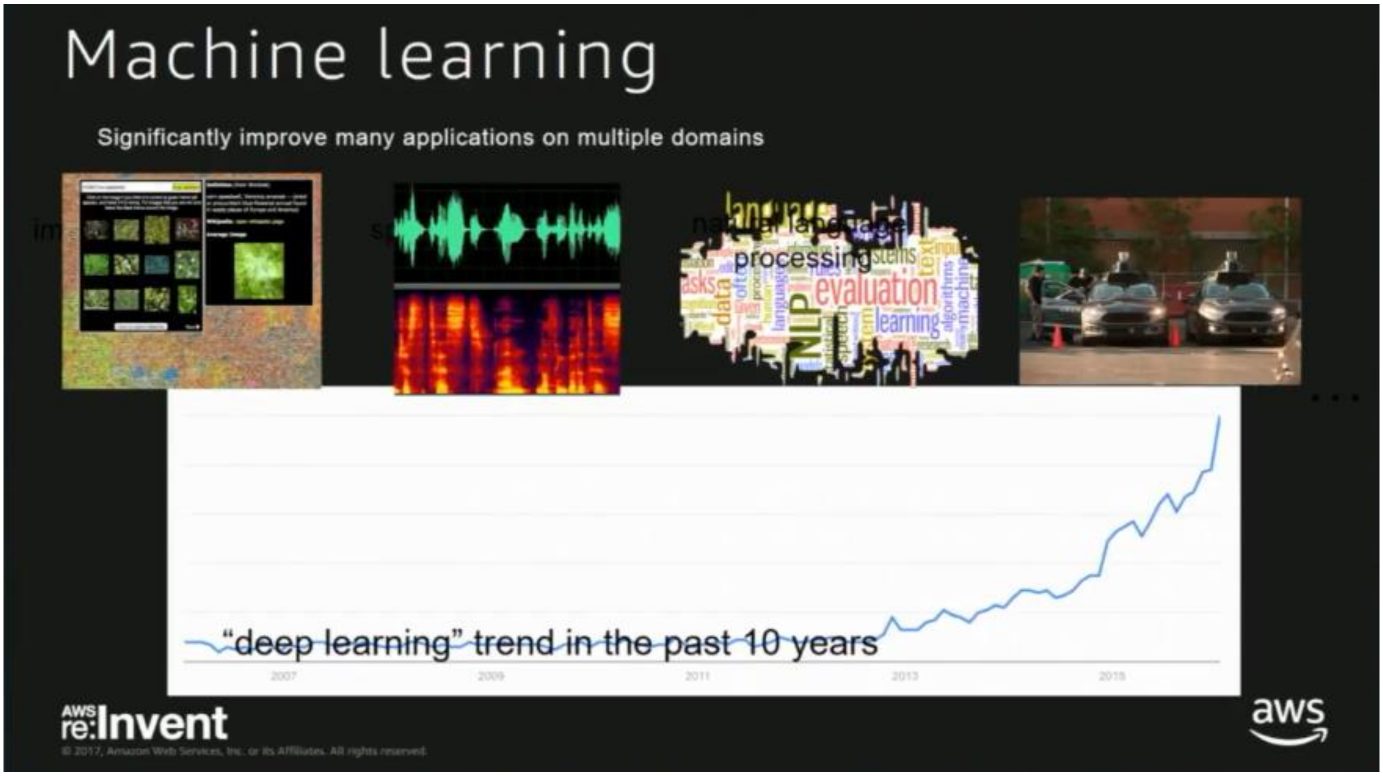
re:Invent
aws

Image recognition is a field of deep learning that uses neural networks to recognize the subject and traits for a given image. In Japan, Cookpad uses Amazon ECS to run an image recognition platform on clusters of GPU-enabled EC2 instances. In this session, hear from Cookpad about the challenges they faced building and scaling this advanced, user-friendly service to ensure high-availability and low-latency for tens of millions of users.



What are we going to talk about

- What is deep learning
- The lifecycle of a deep learning application
- Deploying deep learning functions on Amazon EC2 Container Service (Amazon ECS)
- Cookpad Inc. use case



Machine learning

Significantly improve many applications on multiple domains

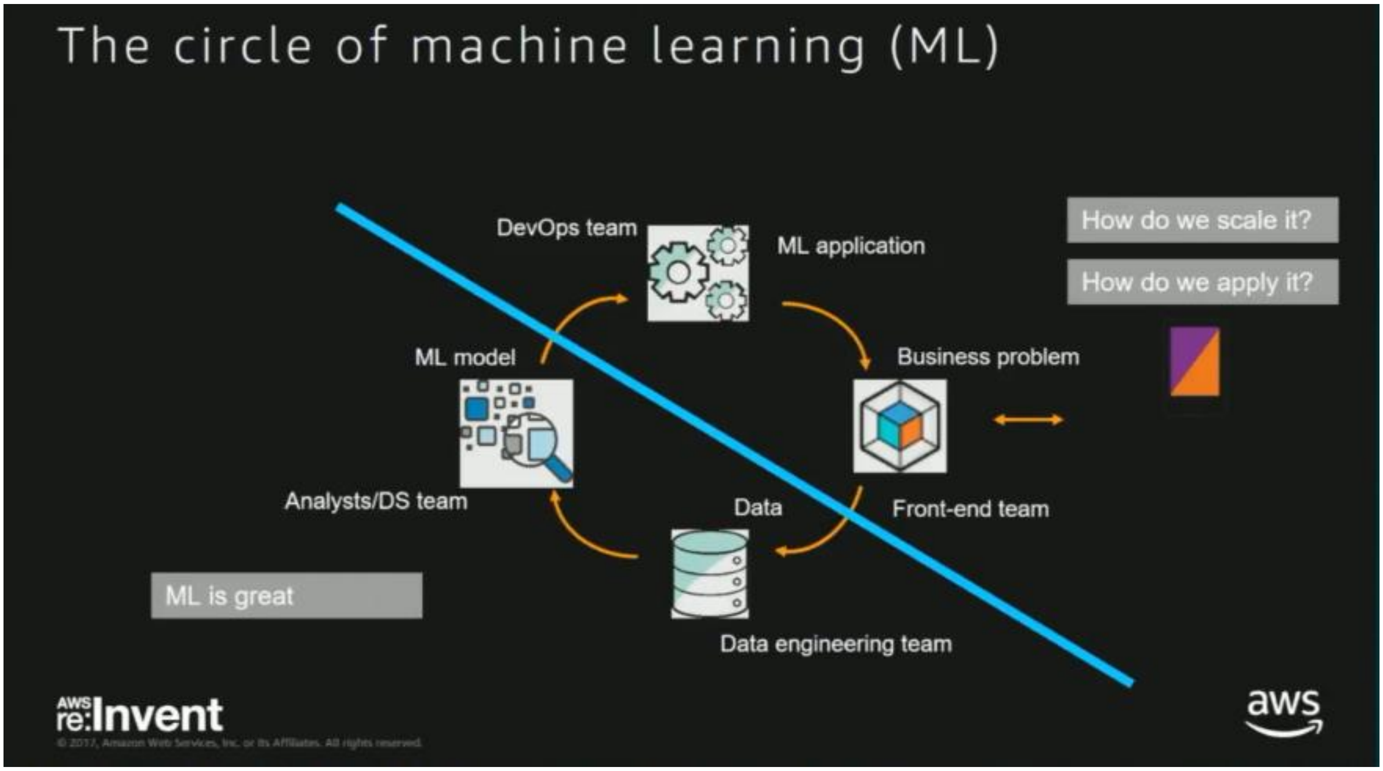
“deep learning” trend in the past 10 years

AWS re:Invent

aws

Machine learning at Amazon

re:Invent



The circle of machine learning (ML)

DevOps team

ML application

How do we scale it?

How do we apply it?

ML model

Business problem

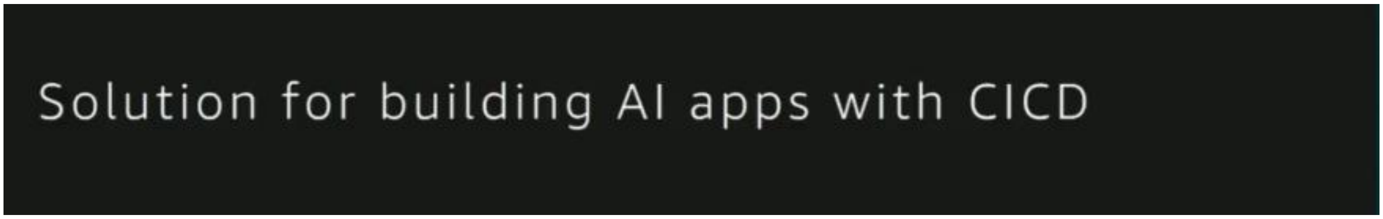
Analysts/DS team

Data

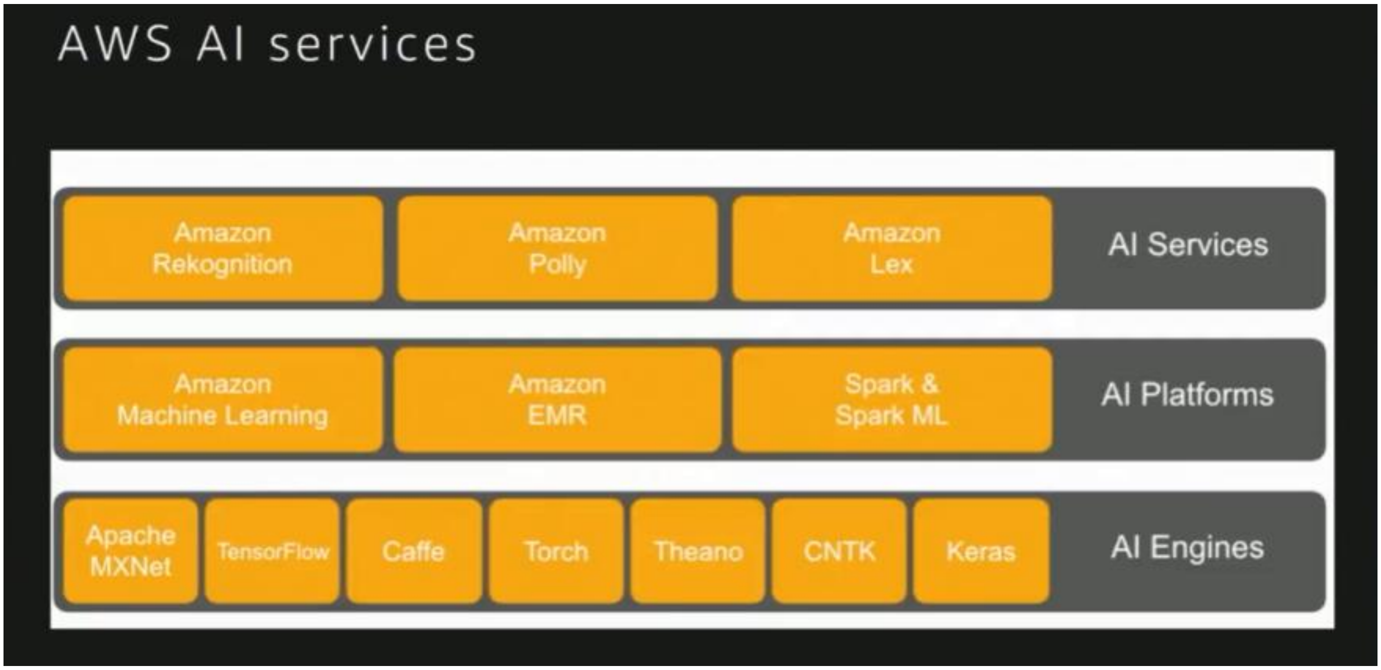
Front-end team

ML is great

Data engineering team

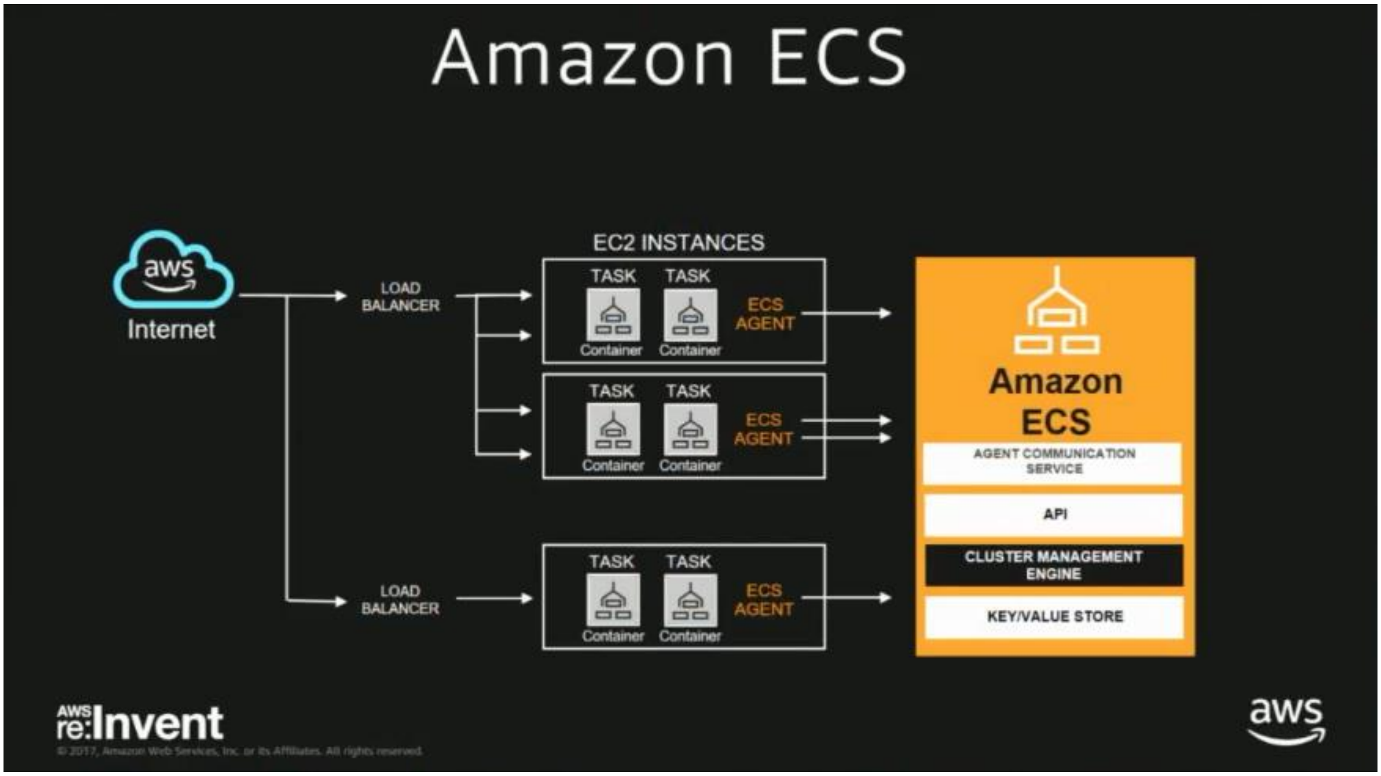


Solution for building AI apps with CICD



AWS AI services

Amazon Rekognition	Amazon Polly	Amazon Lex					AI Services
Amazon Machine Learning	Amazon EMR	Spark & Spark ML					AI Platforms
Apache MXNet	TensorFlow	Caffe	Torch	Theano	CNTK	Keras	AI Engines



Amazon ECS

aws Internet

LOAD BALANCER

LOAD BALANCER

EC2 INSTANCES

TASK TASK

Container Container

ECS AGENT

Amazon ECS

AGENT COMMUNICATION SERVICE

API

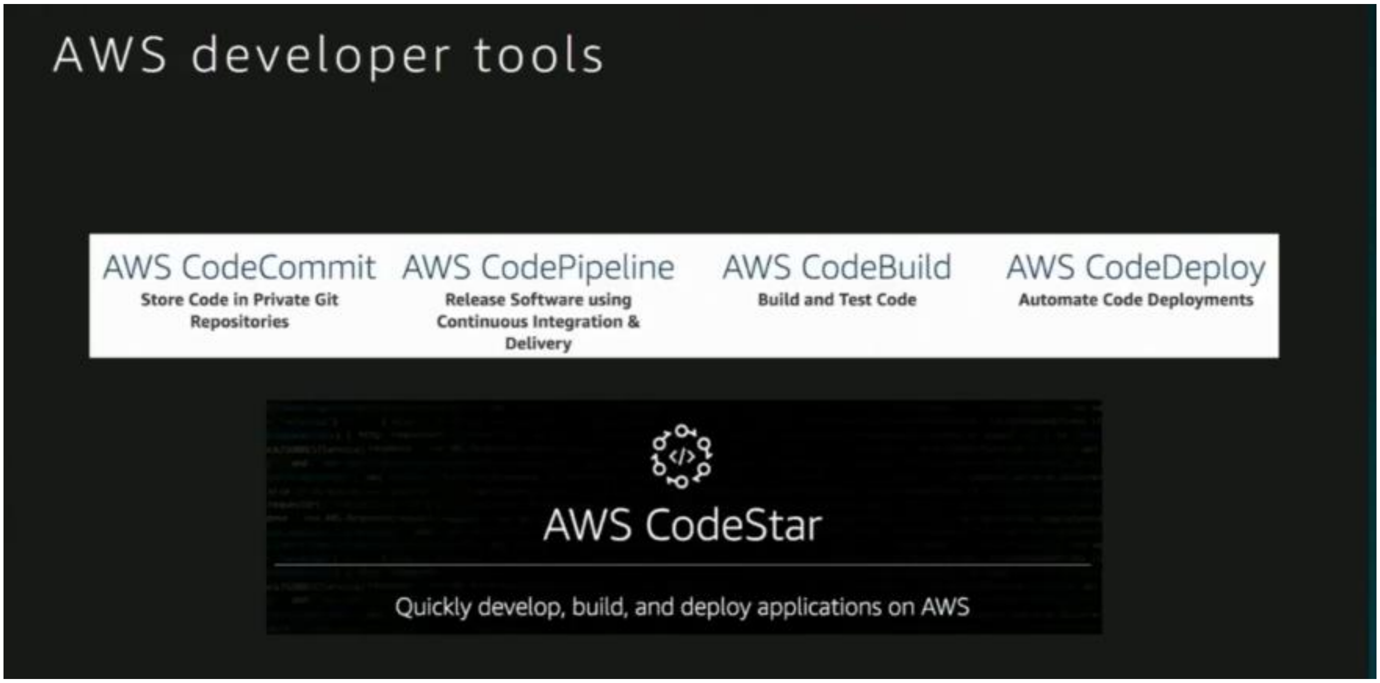
CLUSTER MANAGEMENT ENGINE

KEY/VALUE STORE

AWS re:Invent

aws

The other primitive we need to build our service is a container, we will need to deploy 1000s of containers on 1000s of nodes, we can use ECS for this.



AWS developer tools

AWS CodeCommit
Store Code in Private Git Repositories

AWS CodePipeline
Release Software using Continuous Integration & Delivery

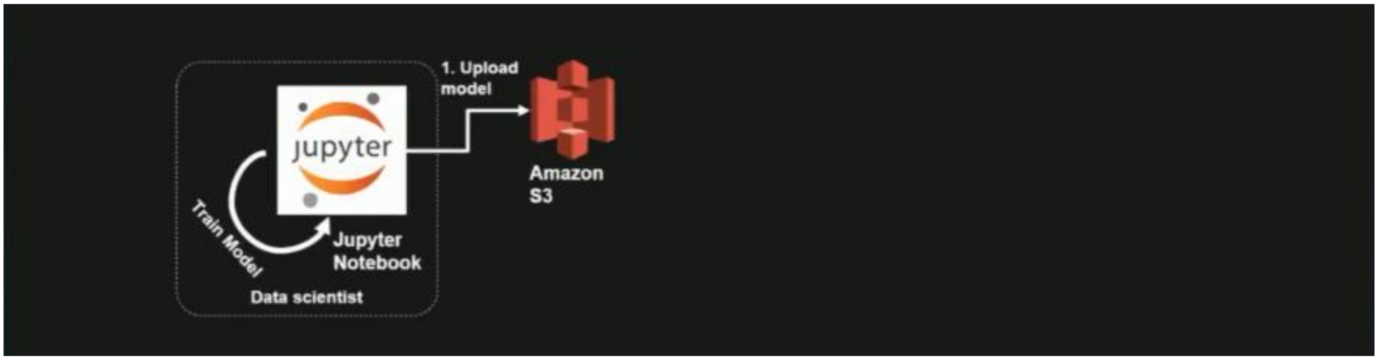
AWS CodeBuild
Build and Test Code

AWS CodeDeploy
Automate Code Deployments

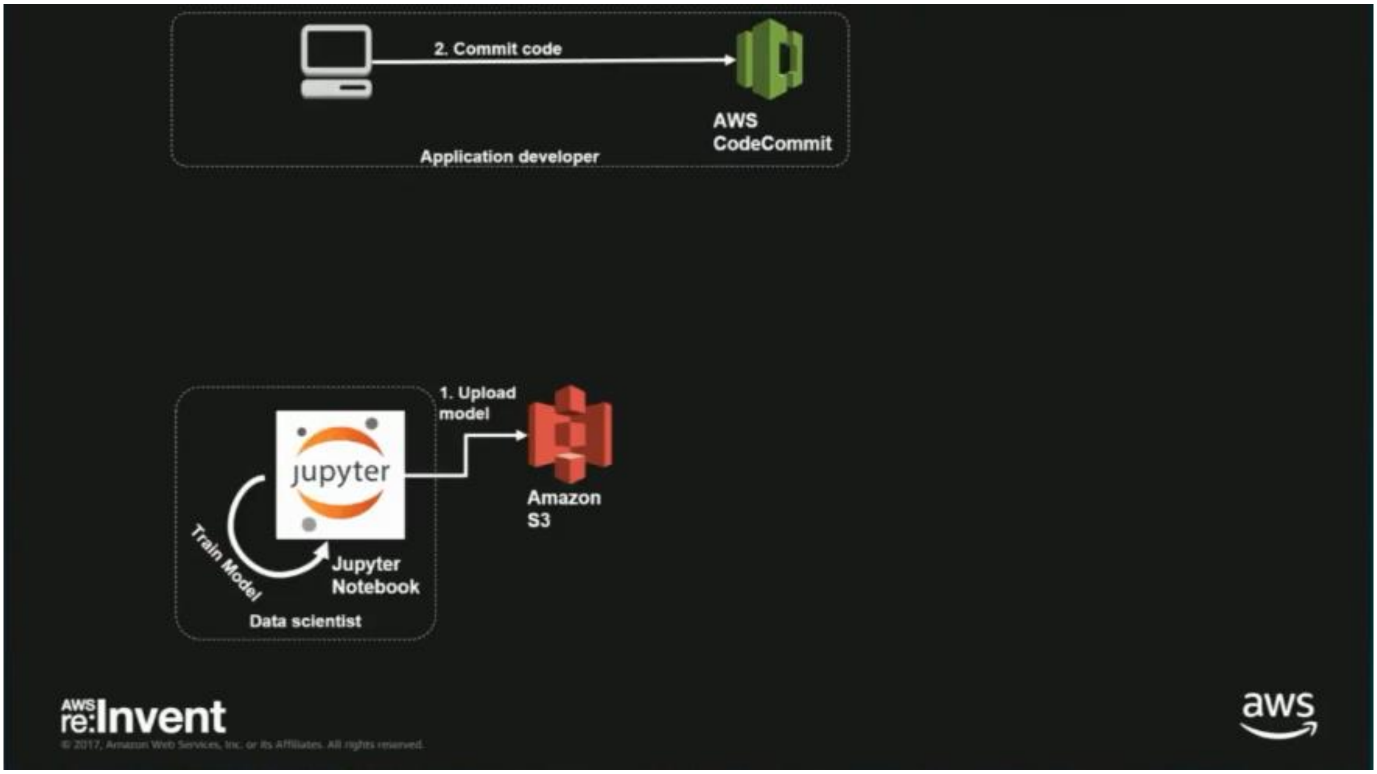
AWS CodeStar

Quickly develop, build, and deploy applications on AWS

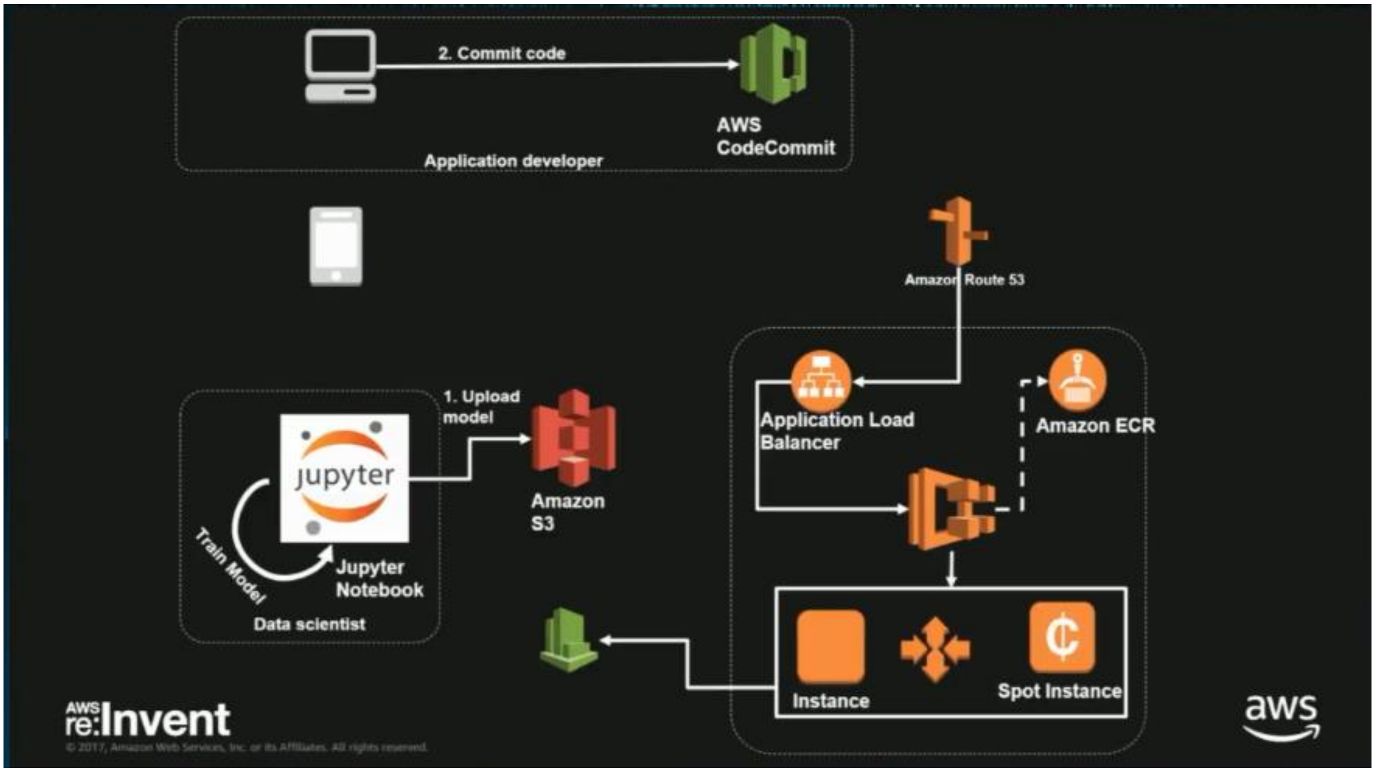
Let’s do this...



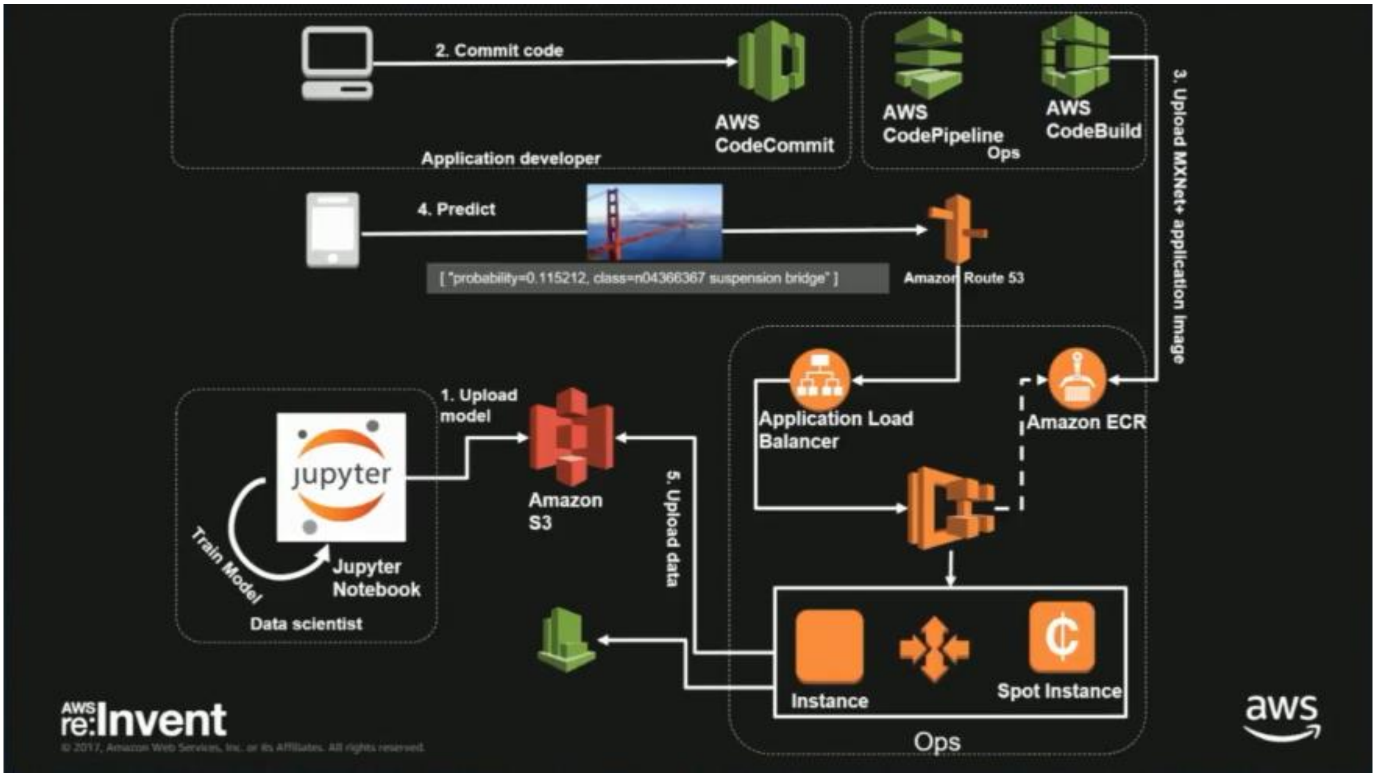
Amazon has published AMIs that you can just click to run in an EC2 instance and you can use for training your models and doing work as a data scientist. We upload our models into S3.



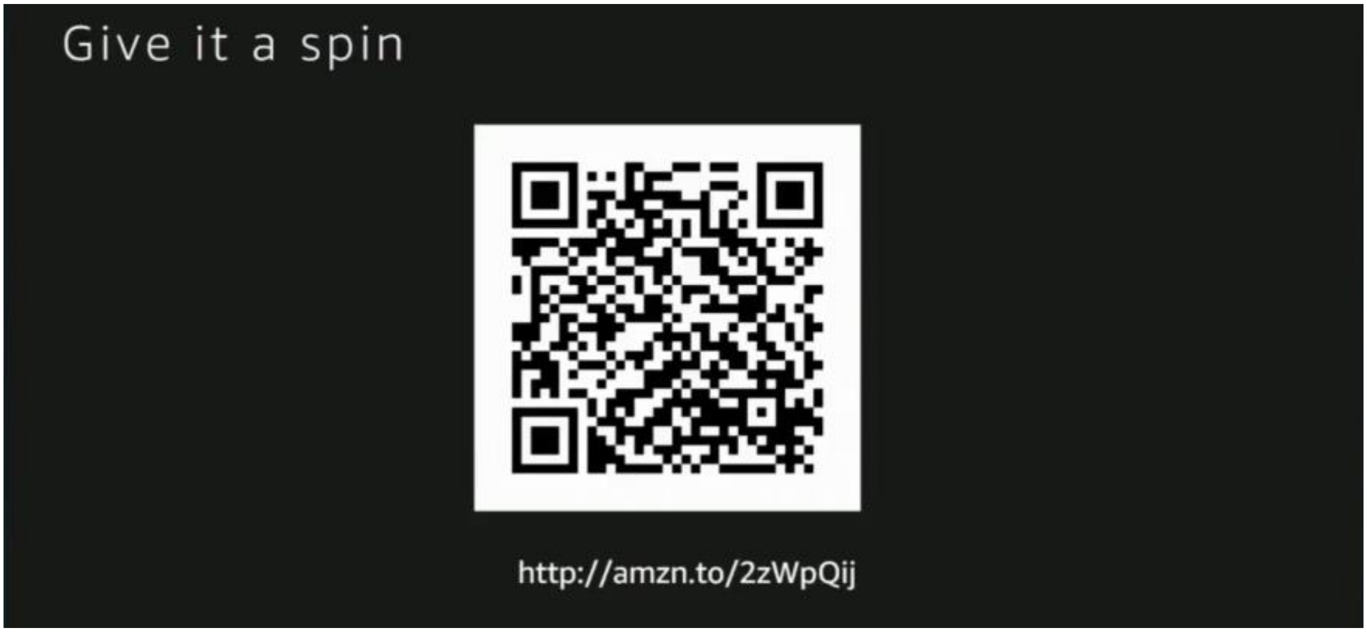
The application developer wants to write code, push it to a repo, build and deploy the app.



We also need a compute to run our workloads, you can download a CF template that can do this for you today

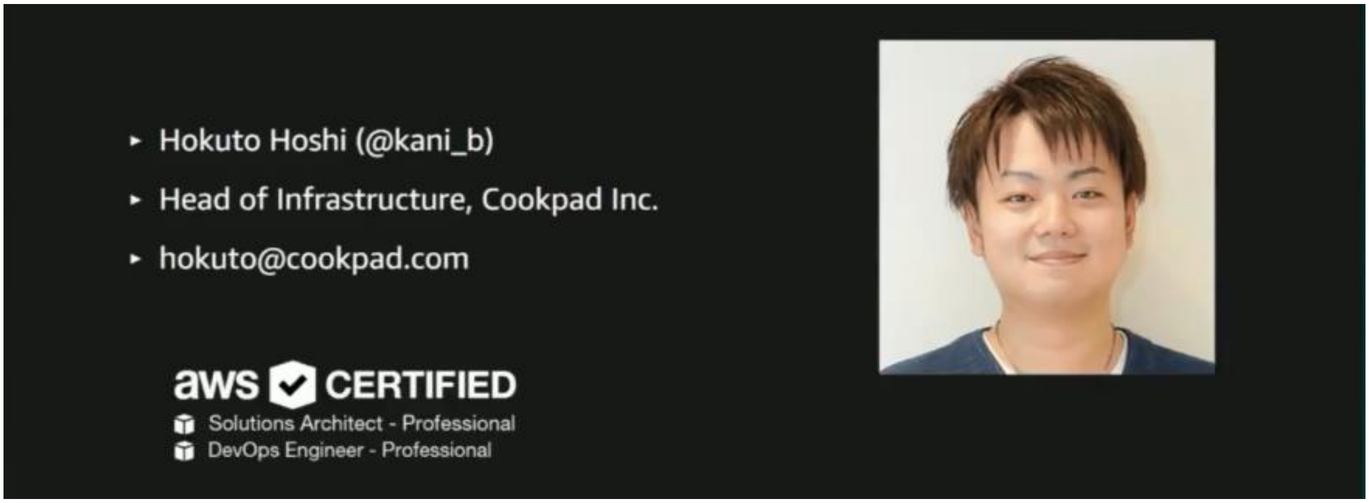


```
[ "probability=0.115212, class=n04366367 suspension bridge" ]
```



Give it a spin

http://amzn.to/2zWpQij



- Hokuto Hoshi (@kani_b)
- Head of Infrastructure, Cookpad Inc.
- hokuto@cookpad.com

aws ✓ CERTIFIED

- Solutions Architect - Professional
- DevOps Engineer - Professional

cookpad
Search Recipes
Create Recipe
COOK IT! SNAP IT! SHARE IT...
クックパッド主催
びっくりレシピ
大作戦
Happy Halloween
詳しくはこちら＞
レシピ
Ayesha Puttanesca
Super Easy Spicy Korean Style Soy Dipping Sauce
Pork and Mushroom stir fry for one person
Salmon and Japanese Mushroom parcel

About Cookpad

- “Make everyday cooking fun!” - Since **1998**
- https://cookpad.com/
- Largest online recipe sharing and search service in Japan

About **60M** Monthly users in Japan

Over **2.7M** user-authored recipes

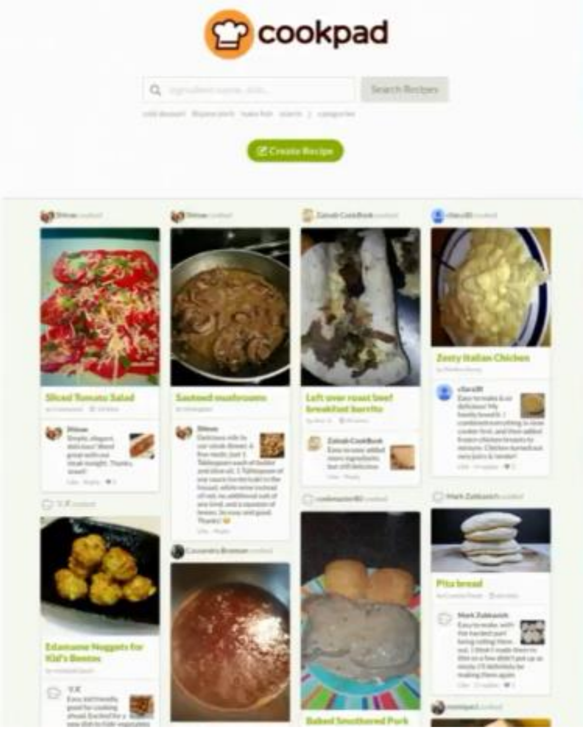
Cookpad is global

- https://cookpad.com/#{your_country_code}
  - us, uk, id, es, fr, br, ae, etc....

67 countries

21 languages

Offices in Japan, UK, Spain, Indonesia etc.



cookpad

Search Recipes

Create Recipe

Sliced Tomato Salad

Sauteed mushrooms

Left over roast beef breakfast burrito

Zesty Italian Chicken

Edamame Nuggets for Kid's Bentos

Pita bread

Baked Smothered Pork

AWS re:Invent

Our infrastructure

150+ developers

9 SREs
9 Machine learning engineers

All-in on AWS since **2011**

powered by aws

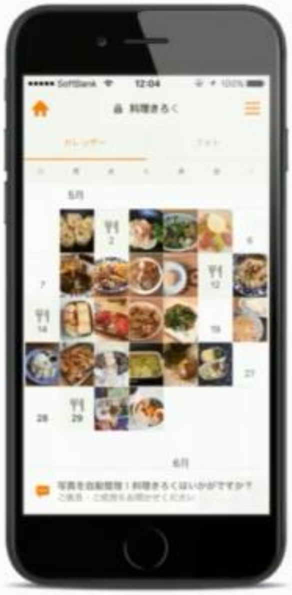
~1,400 EC2 instances
200+ ECS services

Over **2** regions

15,000+ requests/sec

Cooking Log (料理きろく)

Our very first Deep Learning powered feature

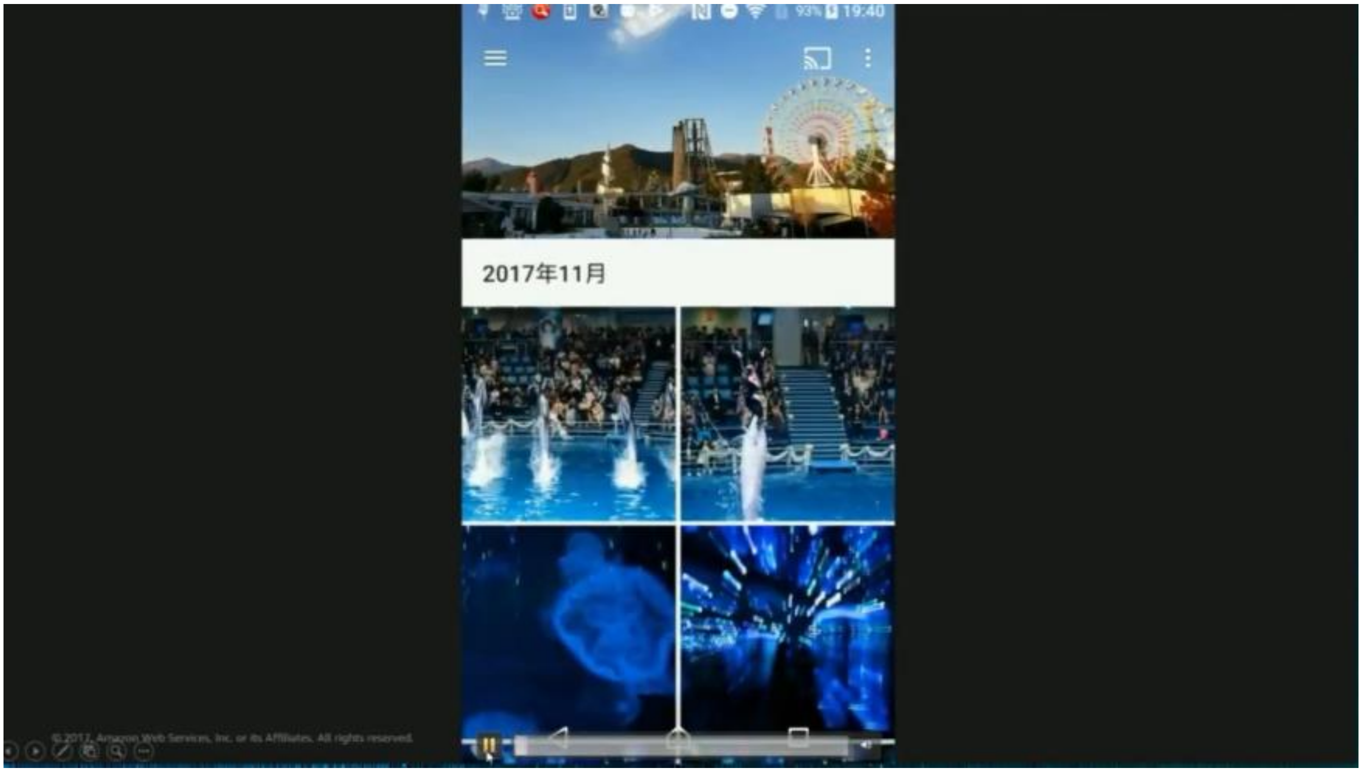


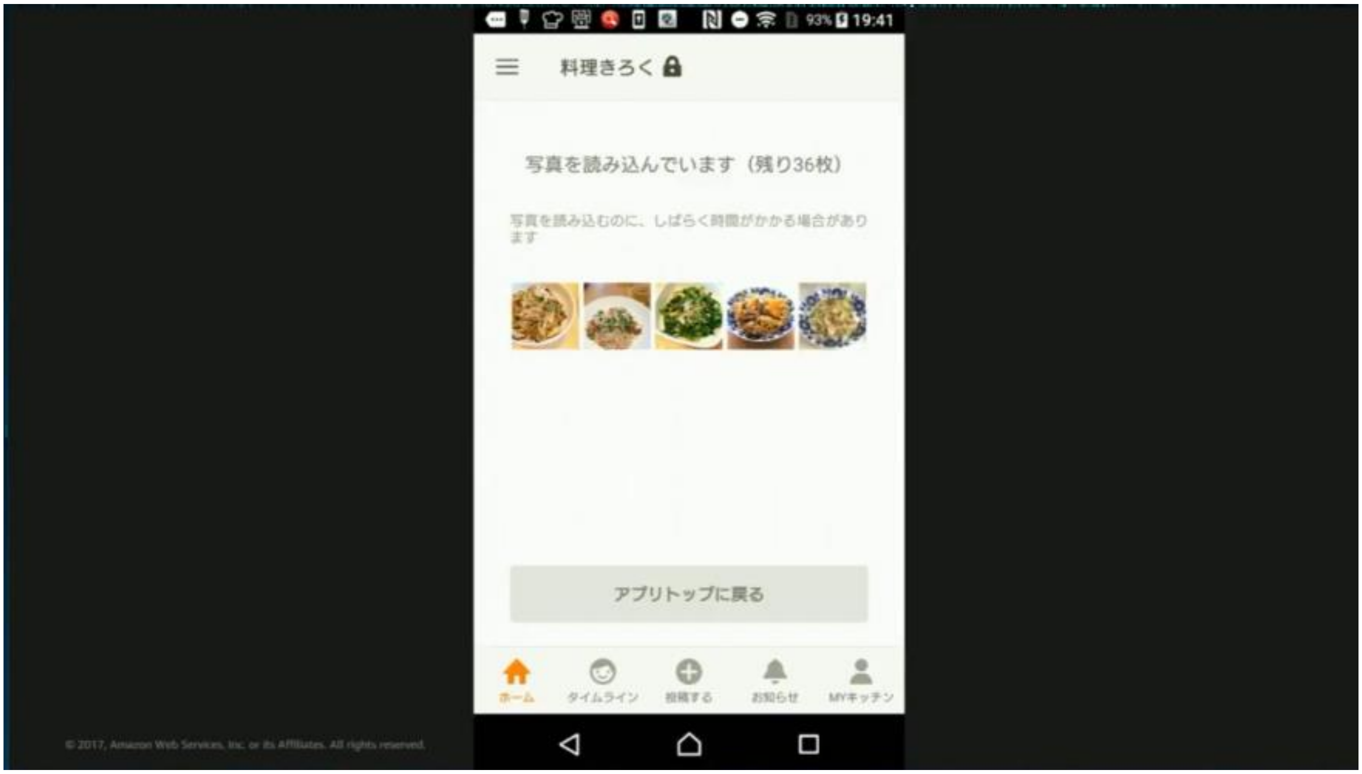
Cooking log
(料理きろく)

Collect "Food Photos" from Camera Roll automatically

Powered by Convolutional Neural Network

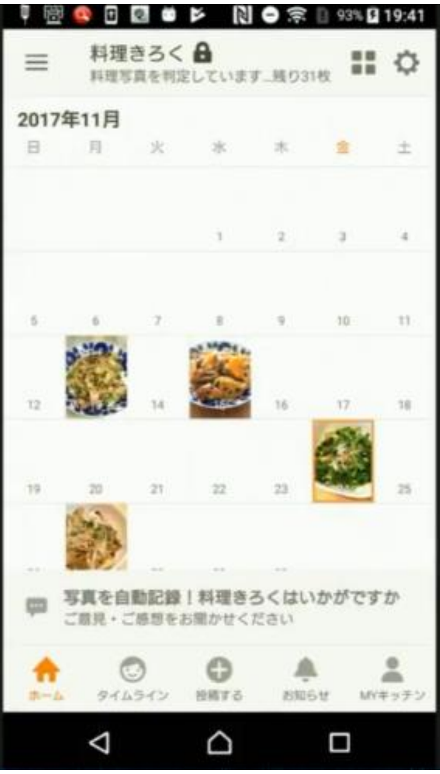
DEMO





写真を読み込んでいます（残り36枚）

写真を読み込むのに、しばらく時間がかかる場合があります



料理きろく

料理写真を判定しています...残り31枚

2017年11月

日	月	火	水	木	金	土
			1	2	3	4
5	6	7	8	9	10	11
12		14		16	17	18
19	20	21	22	23		25

写真を自動記録！料理きろくはいかがですか

ご意見・ご感想をお聞かせください

140,000+
Users

12,000,000+
"food" photos

Our first deep learning feature

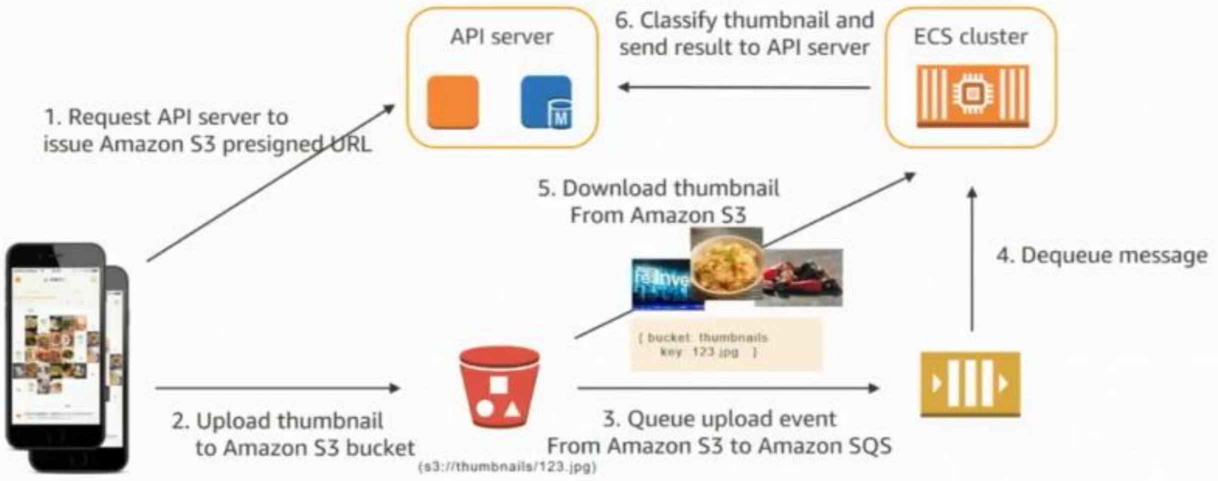
- There were several new challenges
  - Semi real-time image classification in production
  - Different workloads from the rest of web applications
- Especially we needed:
  - Scalable infrastructure for new workloads
  - Environment isolation for new challenge

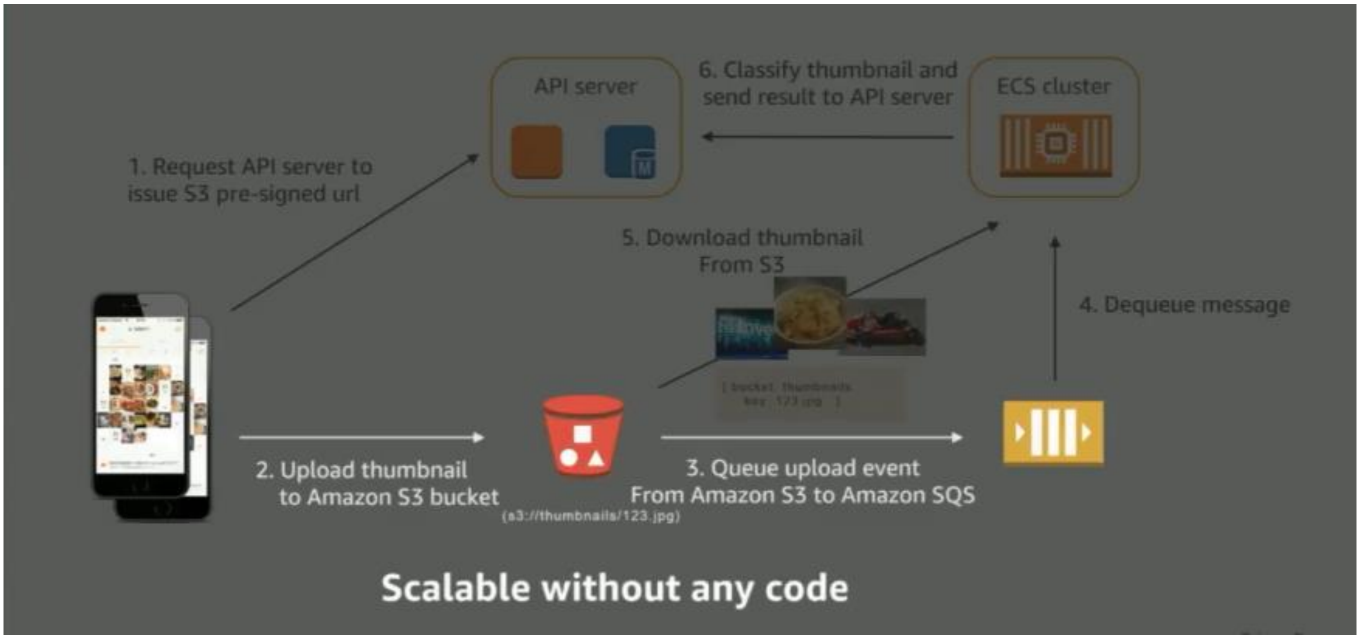
Scalable and asynchronous classification

- What we needed: Scalable infrastructure for massive photo uploading and semi real-time classification
  - Clients send tiny thumbnails after taking photos (difficult to predict traffic)
  - Traffic spikes are coming sometimes (e.g. The TV show introduces our app)
- What we chose: Asynchronous architecture
  - Uploading and classification take time (~ several hundred ms)
    - Synchronous processing with API servers gives users a bad experience
  - Upload directly from clients to Amazon S3 using presigned URL
  - Enable Amazon S3 notification and Amazon SQS for queue of classifi

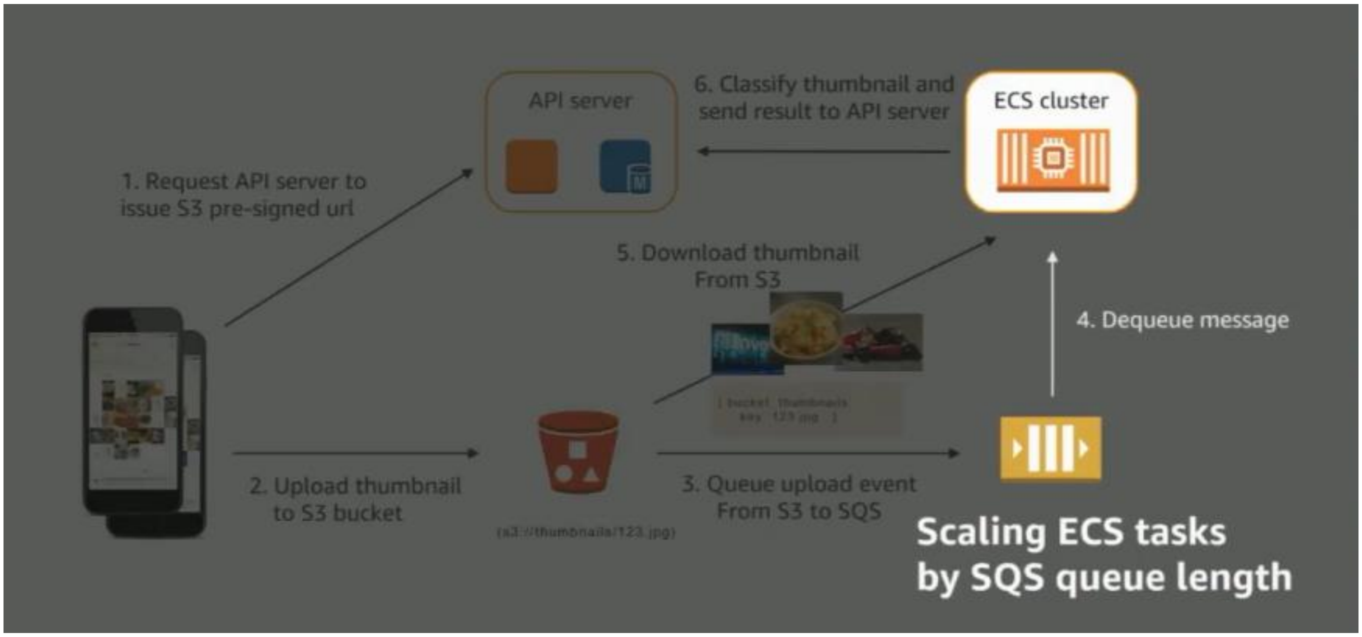
Environment isolation

- What we needed: Isolated environment in production
  - Different languages (Python for Machine Learning, Ruby for web application)
  - Different workloads (It's our first product that uses deep learning)
  - Different hardware (GPU)
- What we chose: Container environment
  - The container ensures runtimes are isolated
    - Language environment, GPU drivers, many configurations
  - Amazon ECS provides managed and scalable Docker environment
  - And we had already used containers on Amazon ECS!
  - We run all classification in Amazon ECS cluster on g2.xlarge

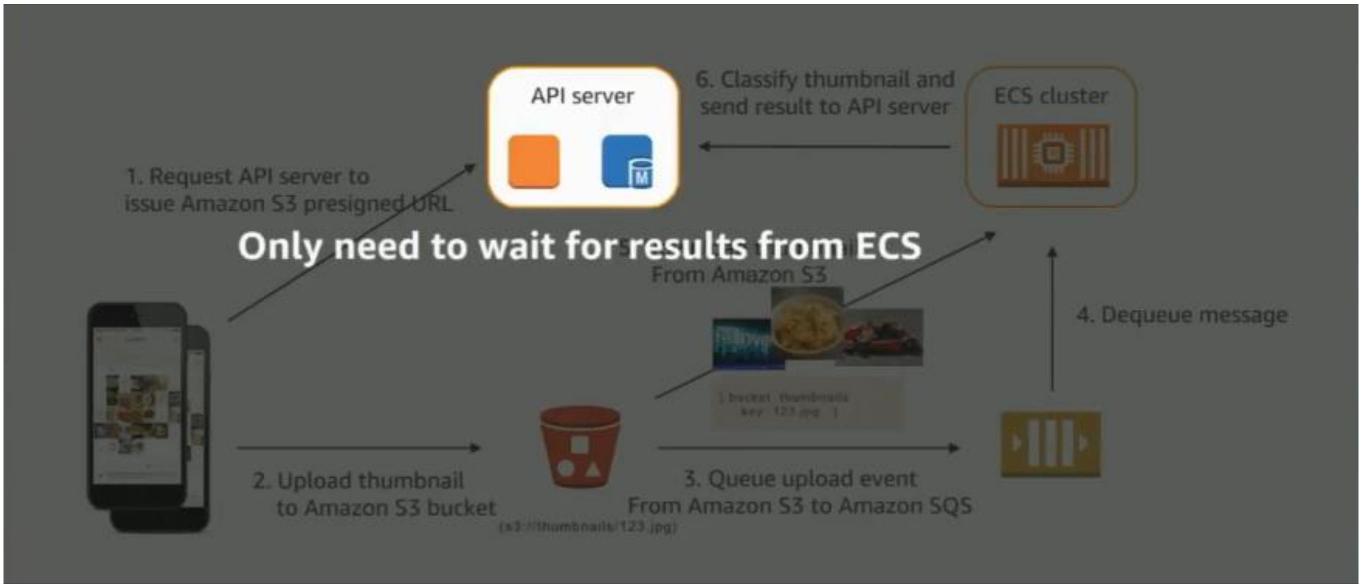

API server
ECS cluster
6. Classify thumbnail and send result to API server
1. Request API server to issue Amazon S3 presigned URL
5. Download thumbnail From Amazon S3
4. Dequeue message
[bucket: thumbnails key: 123.jpg]
2. Upload thumbnail to Amazon S3 bucket
(s3://thumbnails/123.jpg)
3. Queue upload event From Amazon S3 to Amazon SQS


API server
6. Classify thumbnail and send result to API server
ECS cluster
1. Request API server to issue S3 pre-signed url
5. Download thumbnail From S3
4. Dequeue message
{ bucket: thumbnails
key: 123.jpg }
2. Upload thumbnail to Amazon S3 bucket
(s3://thumbnails/123.jpg)
3. Queue upload event From Amazon S3 to Amazon SQS

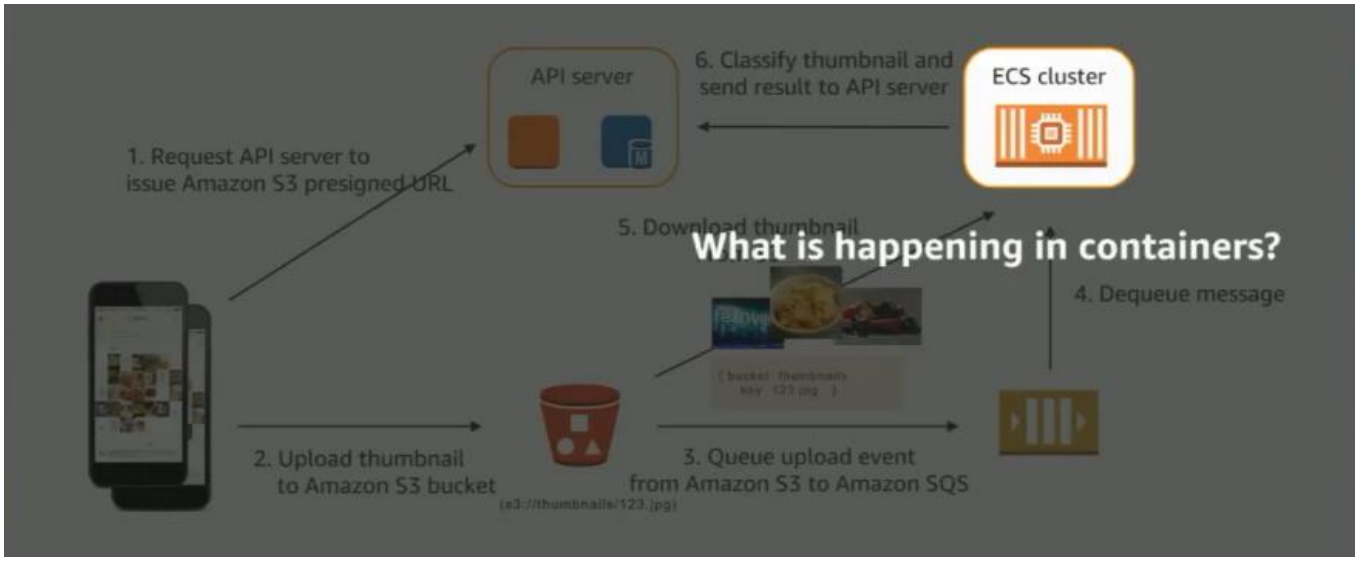
Scalable without any code


API server
6. Classify thumbnail and send result to API server
ECS cluster
1. Request API server to issue S3 pre-signed url
5. Download thumbnail From S3
4. Dequeue message
{ bucket: thumbnails
key: 123.jpg }
2. Upload thumbnail to S3 bucket
(s3://thumbnails/123.jpg)
3. Queue upload event From S3 to SQS

Scaling ECS tasks by SQS queue length


API server
6. Classify thumbnail and send result to API server
ECS cluster
1. Request API server to issue Amazon S3 presigned URL
From Amazon S3
4. Dequeue message
{ bucket: thumbnails
key: 123.jpg }
2. Upload thumbnail to Amazon S3 bucket
(s3://thumbnails/123.jpg)
3. Queue upload event From Amazon S3 to Amazon SQS

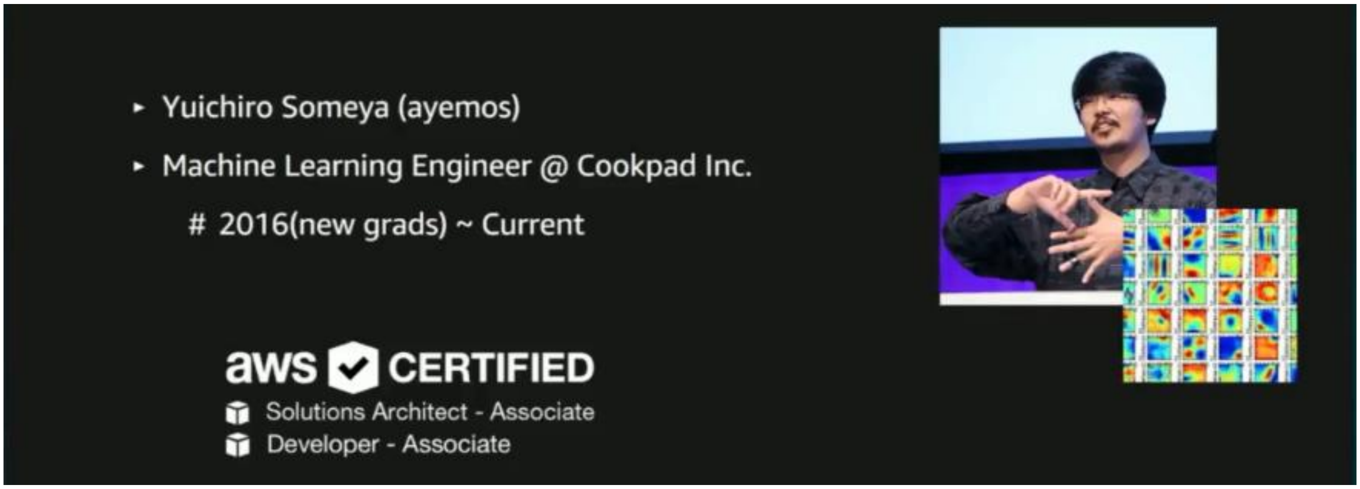
Only need to wait for results from ECS


API server
6. Classify thumbnail and send result to API server
ECS cluster
1. Request API server to issue Amazon S3 presigned URL
5. Download thumbnail
4. Dequeue message
{ bucket: thumbnails key: 123.jpg }
2. Upload thumbnail to Amazon S3 bucket
(s3://thumbnails/123.jpg)
3. Queue upload event from Amazon S3 to Amazon SQS

What is happening in containers?

Machine learning and infrastructure

Infrastructure to accelerate our machine learning projects.

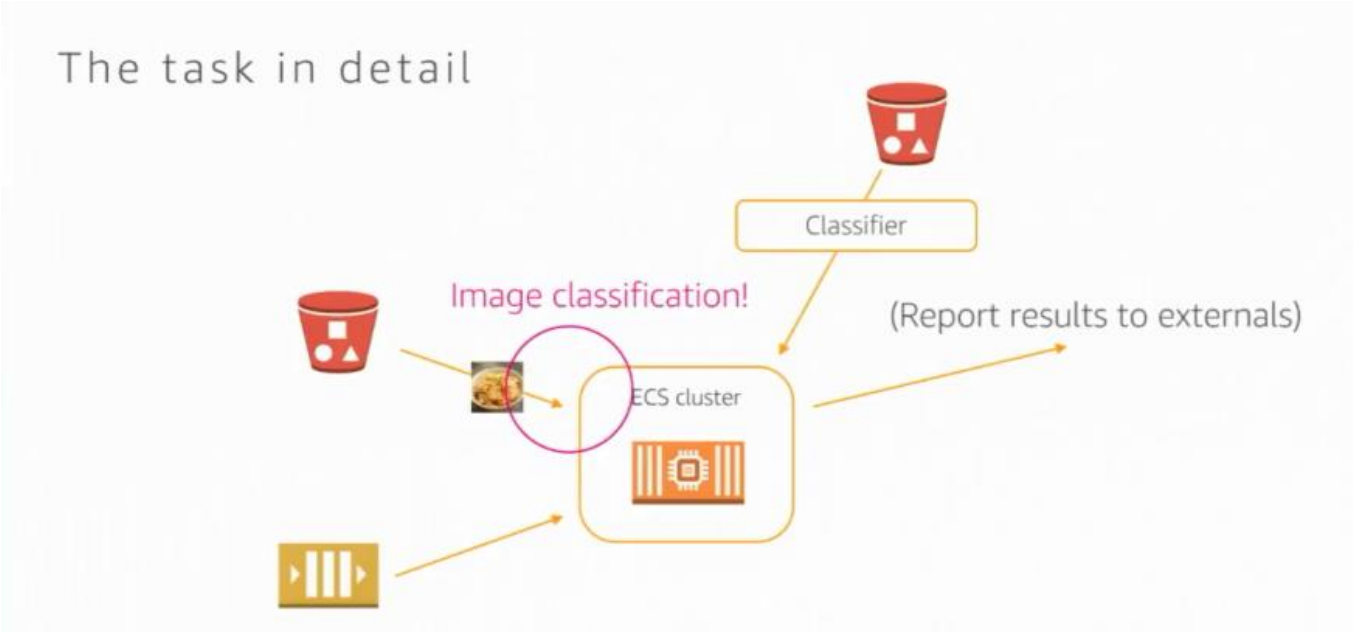


- Yuichiro Someya (ayemos)
- Machine Learning Engineer @ Cookpad Inc.
  - # 2016(new grads) ~ Current

aws CERTIFIED

Solutions Architect - Associate

Developer - Associate



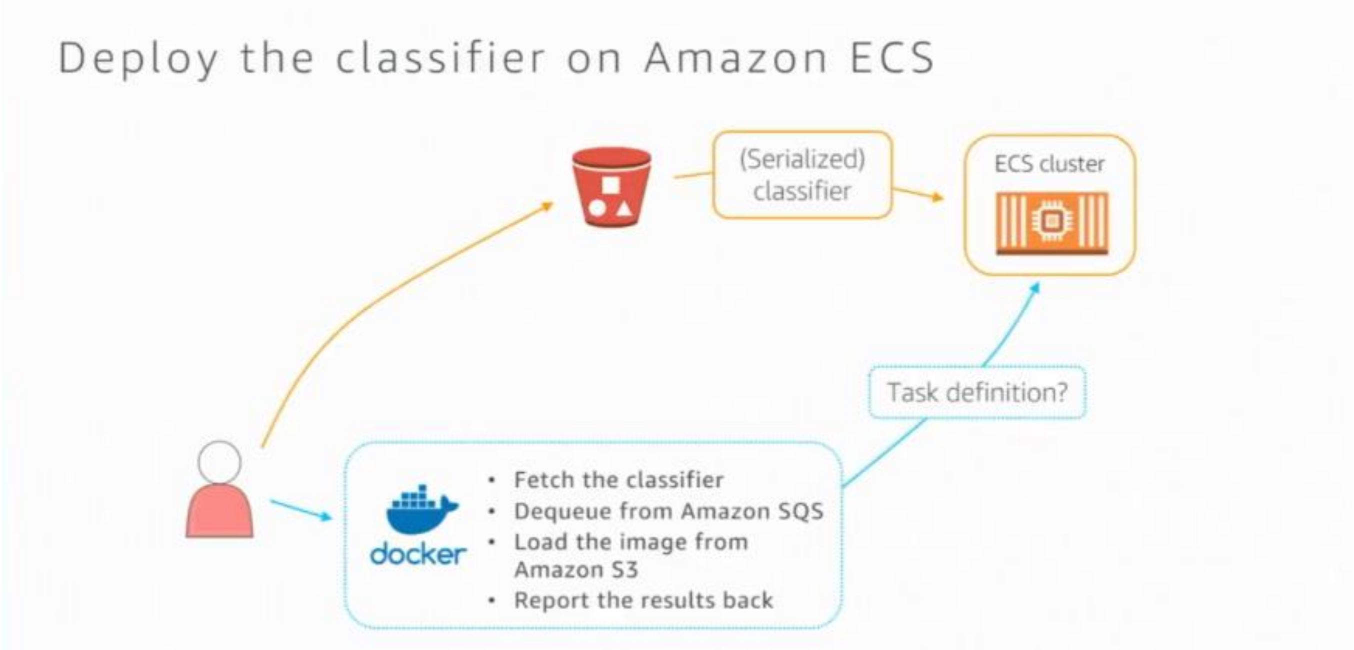
The task in detail

Classifier

Image classification!

(Report results to externals)

ECS cluster



Deploy the classifier on Amazon ECS

(Serialized) classifier

ECS cluster

Task definition?

docker

- Fetch the classifier
- Dequeue from Amazon SQS
- Load the image from Amazon S3
- Report the results back

Democratize task definitions

- hako: https://github.com/eagletmt/hako/
  - Container deploy tool (Amazon ECS compatible)
  - Use yaml as definition file format

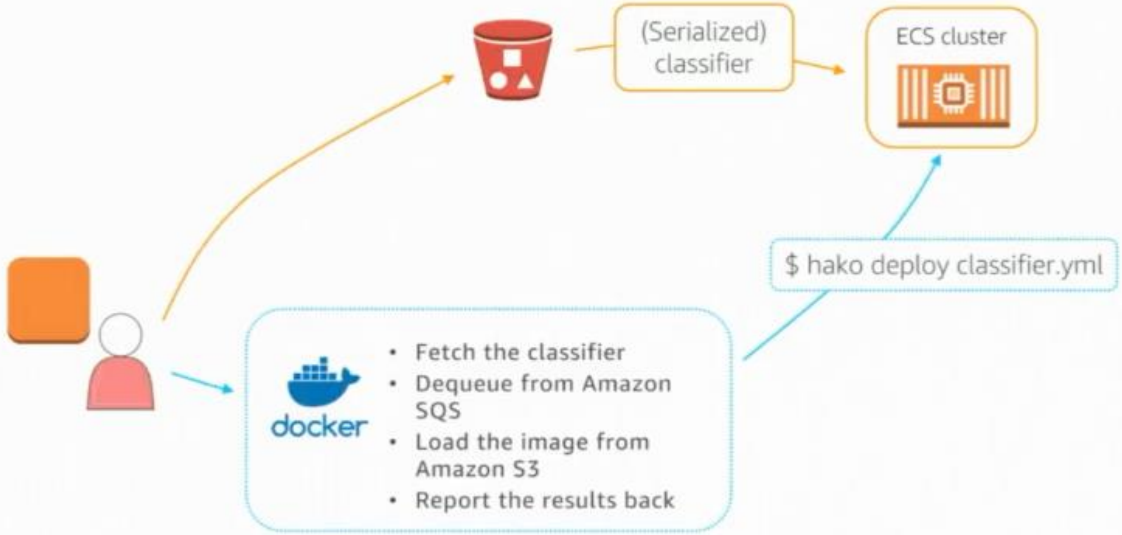
- Each developer writes app.yml and sends Pull request
- **200+** applications are at work

```
scheduler:
  type: ecs
  region: ap-northeast-1
  cluster: hako-production-g2
  desired_count: 1
app:
  image: food-photo-classifier
  cpu: 128
  memory: 3072
  memory_reservation: 2048
  env:
   AWS_REGION: ap-northeast-1
    COOKPADNET_ENV: production
    ...
```

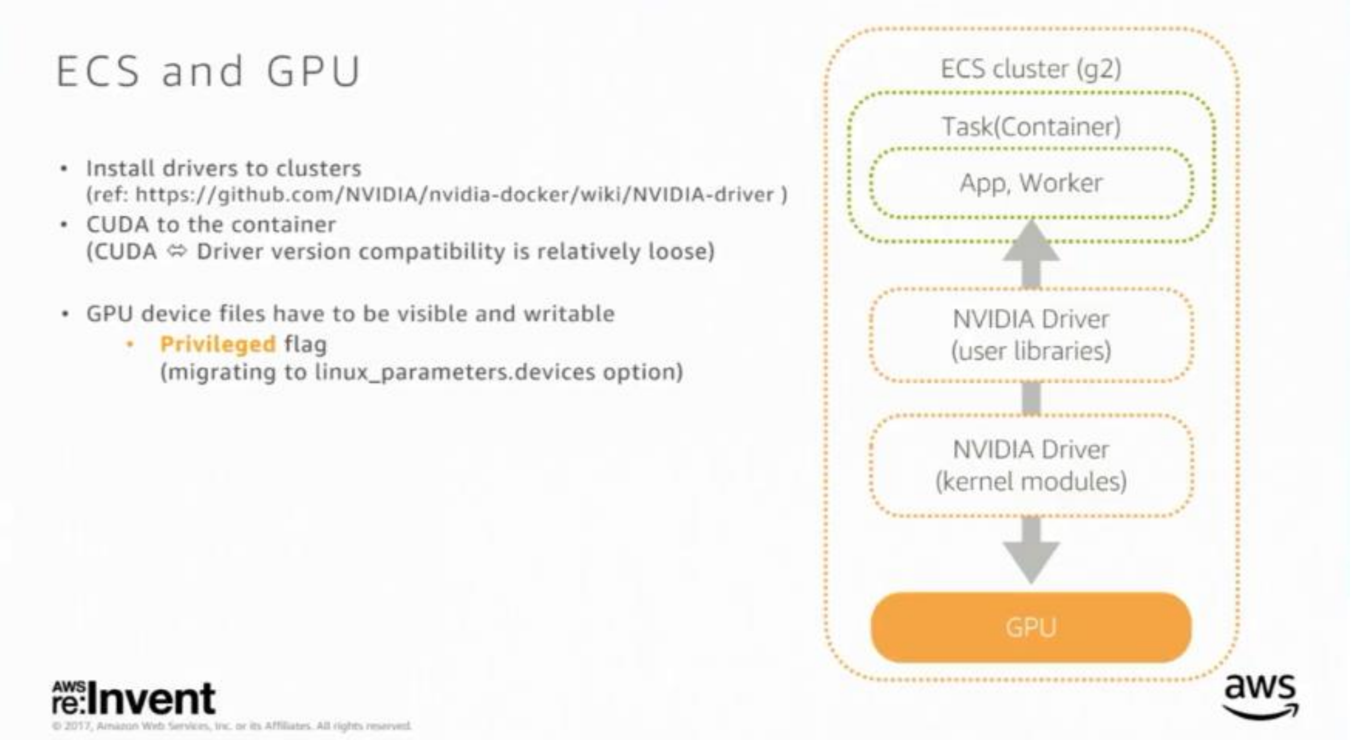
Why we're using hako

- \`hako\` behaves as abstract operation layer over Amazon ECS (or other docker manager)
  - Higher-level operations: Deploy/Rollback/Stop/Remove
  - Manages *secret* environment variables
  - Pluggable pre/post development operations as \`scripts\`
    - Operations like DNS settings, Consul registrations, and so on
- We want each developer can deploy tasks on Amazon ECS individually.
  - \`hako\` handles Infra/SRE work around Amazon ECS.

Deploy the model on Amazon ECS



- Fetch the classifier
- Dequeue from Amazon SQS
- Load the image from Amazon S3
- Report the results back



ECS and GPU

- Install drivers to clusters
  (ref: https://github.com/NVIDIA/nvidia-docker/wiki/NVIDIA-driver)
- CUDA to the container
  (CUDA ⇔ Driver version compatibility is relatively loose)

- GPU device files have to be visible and writable
  - **Privileged** flag
    (migrating to linux_parameters.devices option)

ECS cluster (g2)

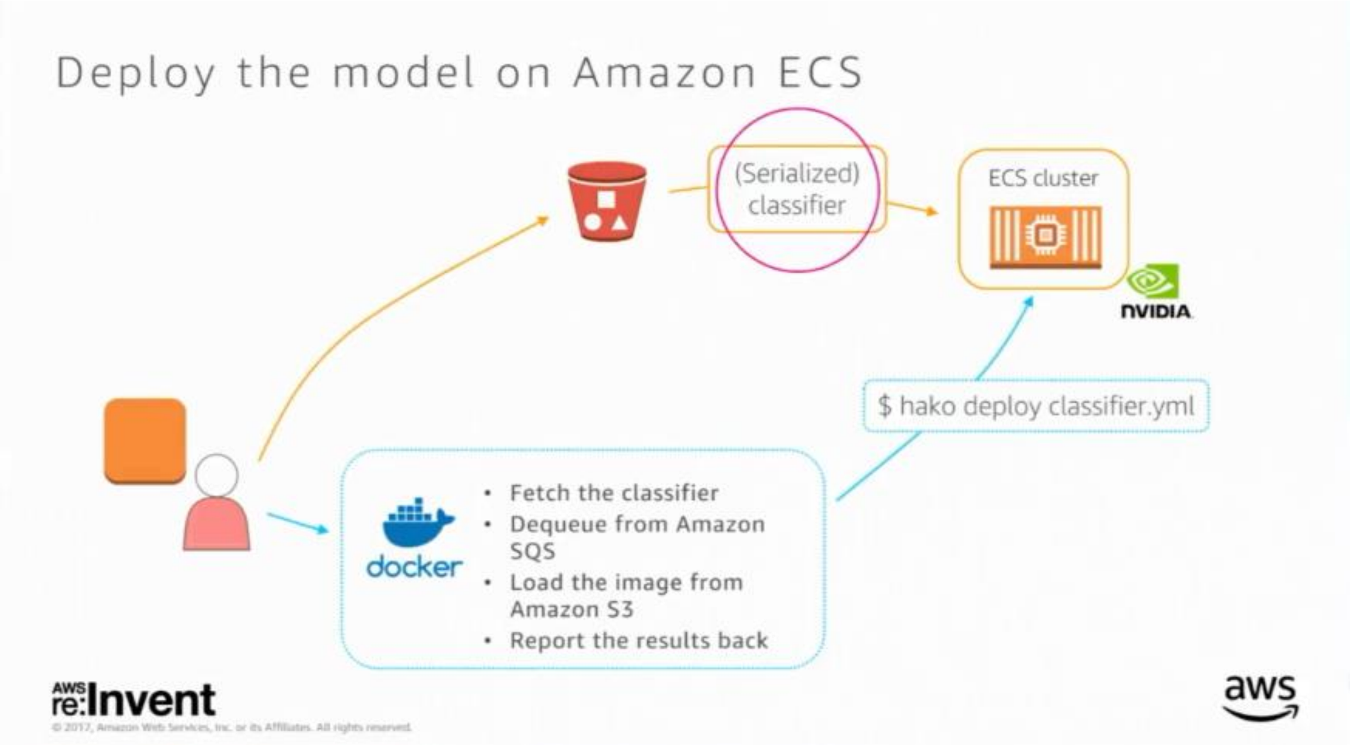
Task(Container)

App, Worker

NVIDIA Driver
(user libraries)

NVIDIA Driver
(kernel modules)

GPU



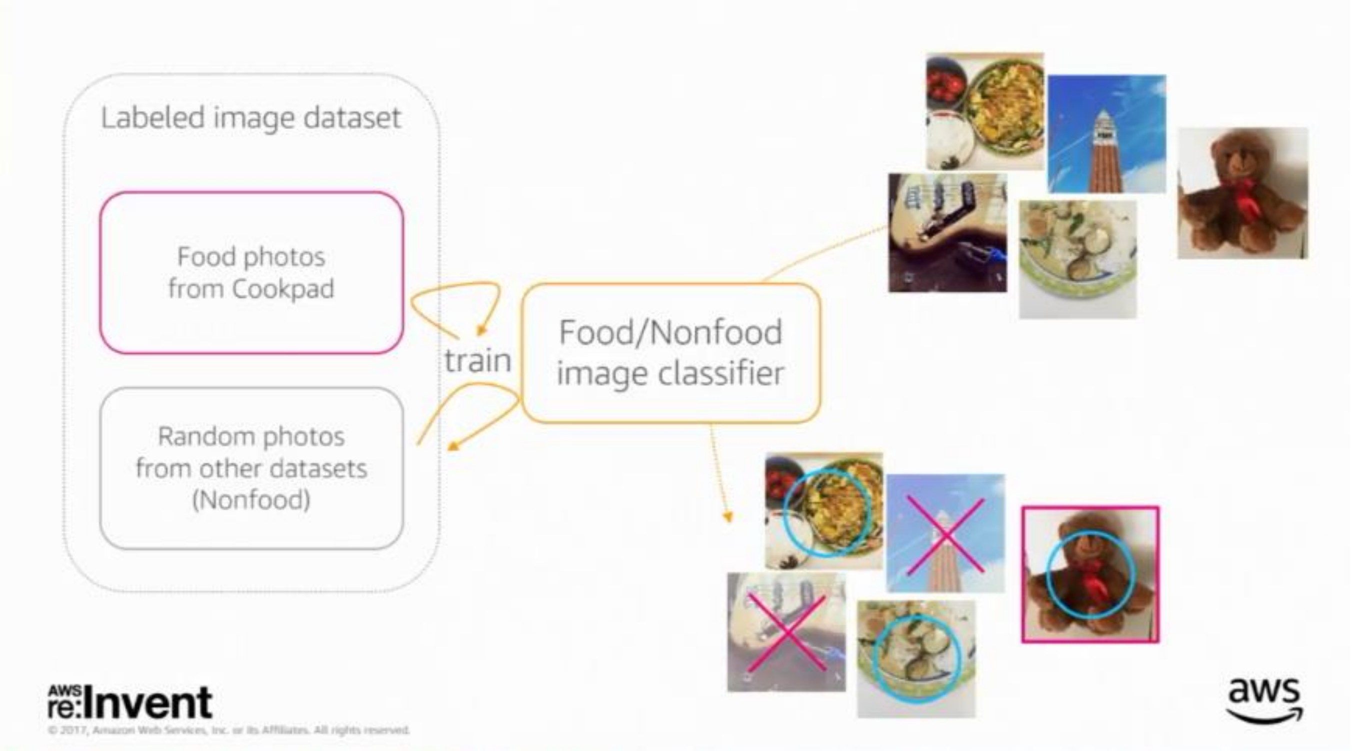
Deploy the model on Amazon ECS

(Serialized)
classifier

ECS cluster

$ hako deploy classifier.yml

- Fetch the classifier
- Dequeue from Amazon SQS
- Load the image from Amazon S3
- Report the results back



Labeled image dataset

Food photos from Cookpad

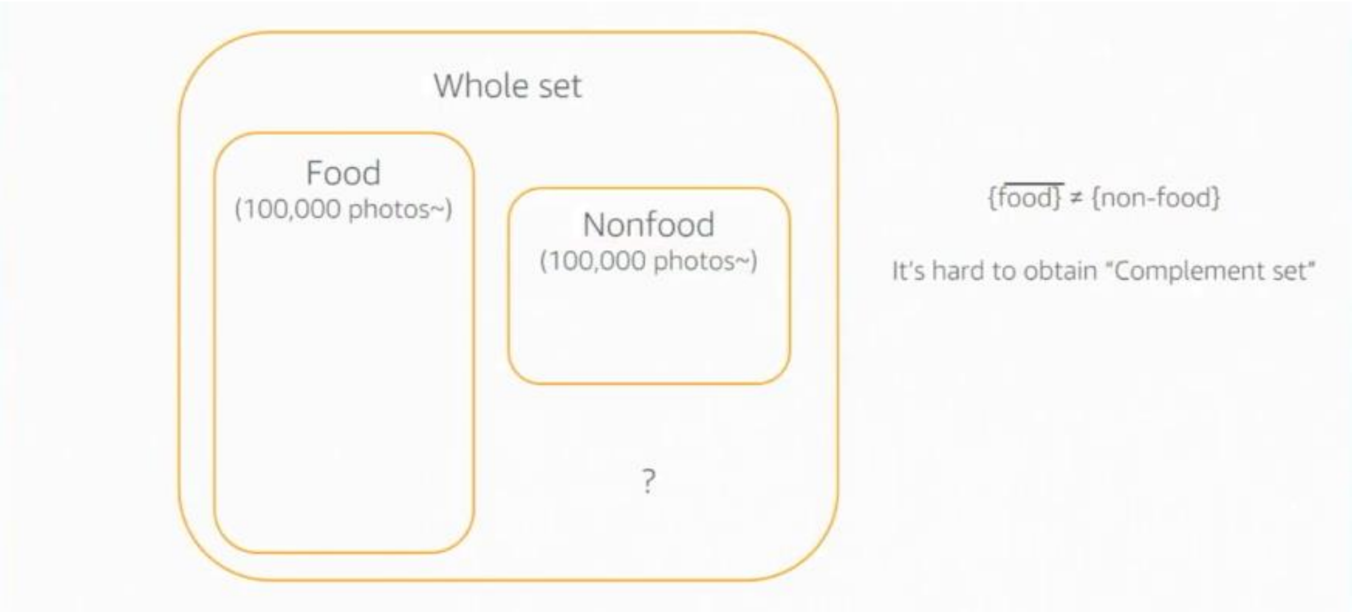
Random photos from other datasets (Nonfood)

train

Food/Nonfood image classifier

AWS re:Invent

aws



Whole set

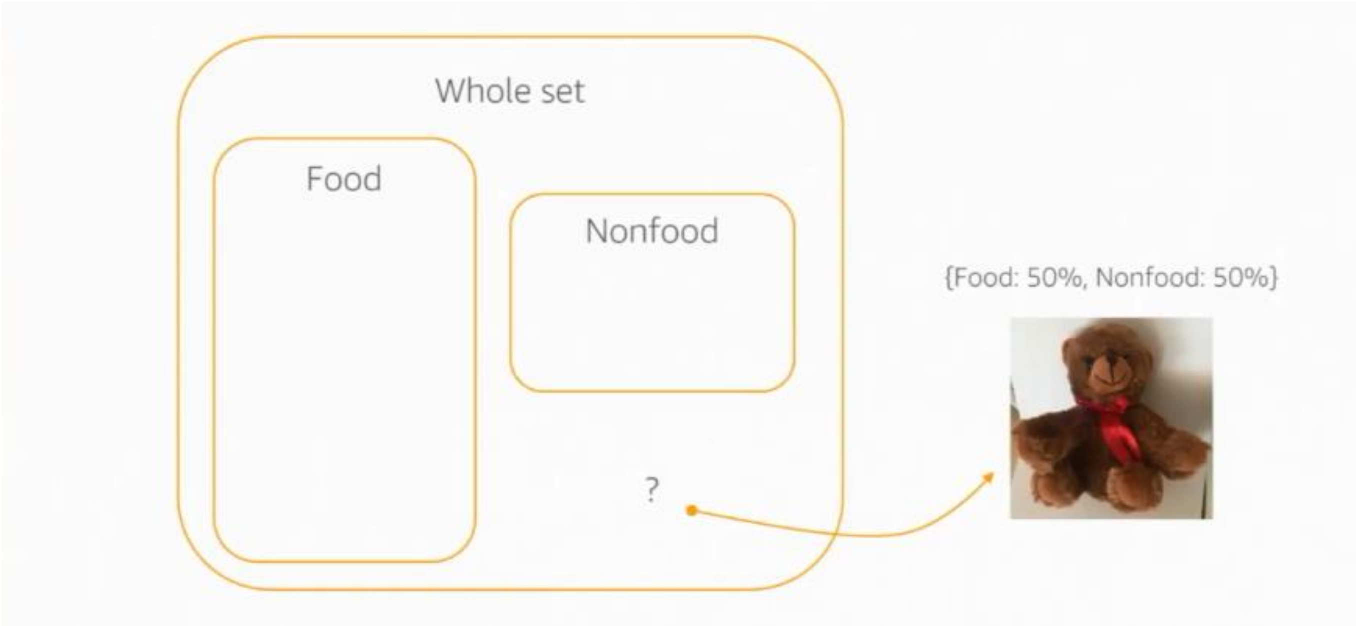
Food (100,000 photos~)

Nonfood (100,000 photos~)

?

$\overline{\{\text{food}\}} \neq \{\text{non-food}\}$

It's hard to obtain "Complement set"



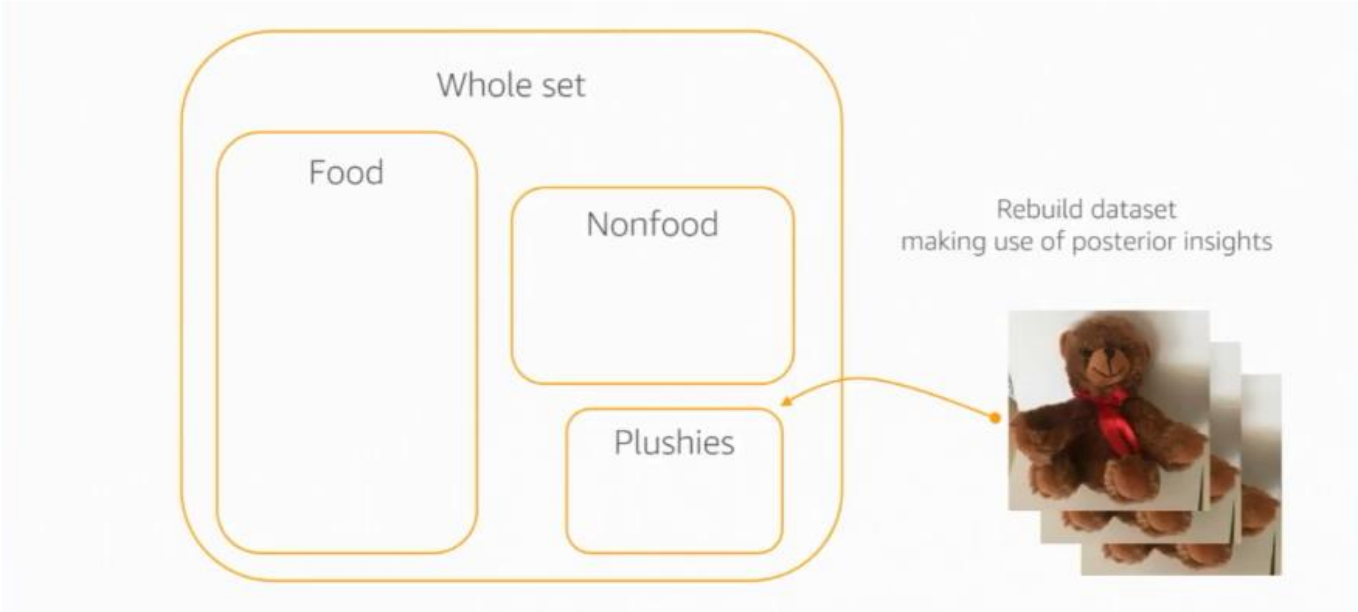
Whole set

Food

Nonfood

?

{Food: 50%, Nonfood: 50%}



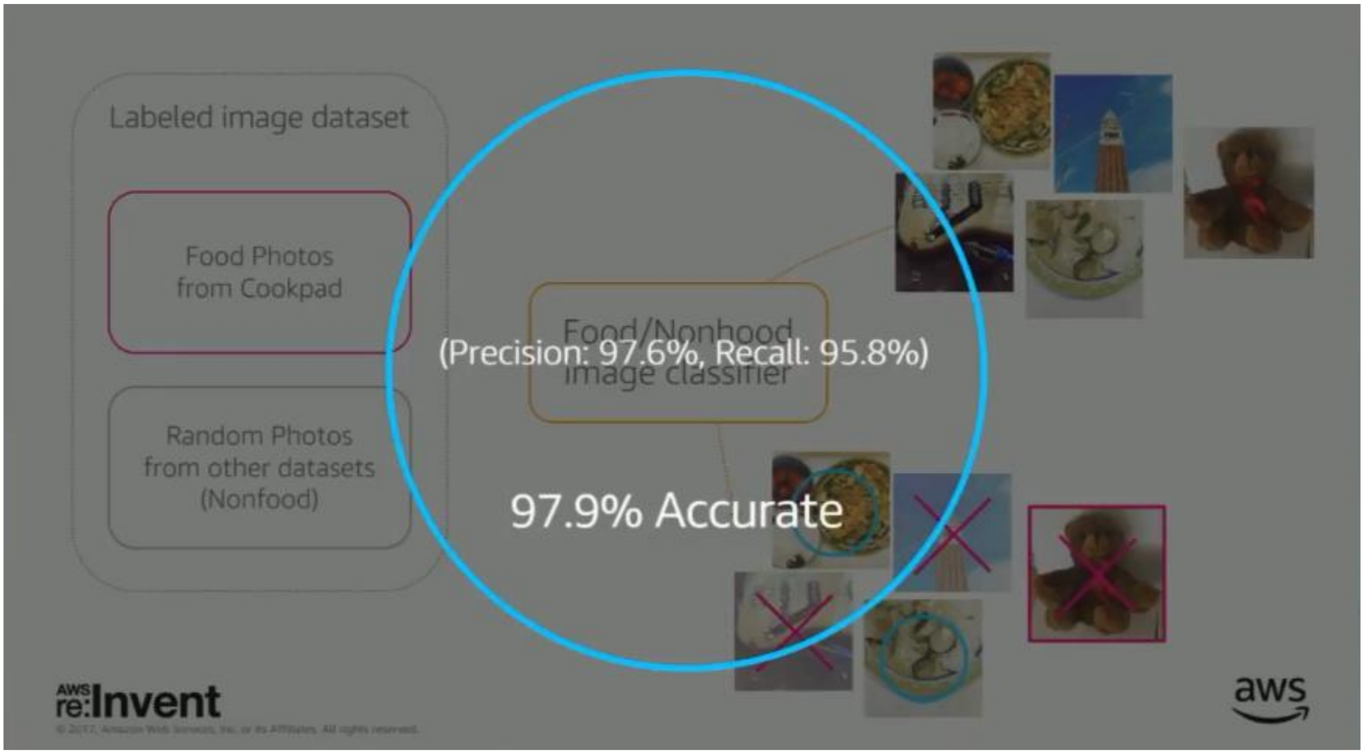
Whole set

Food

Nonfood

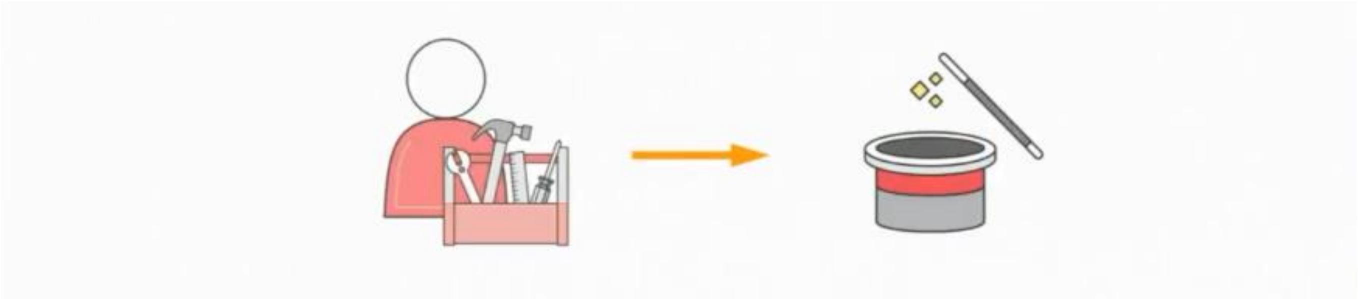
Plushies

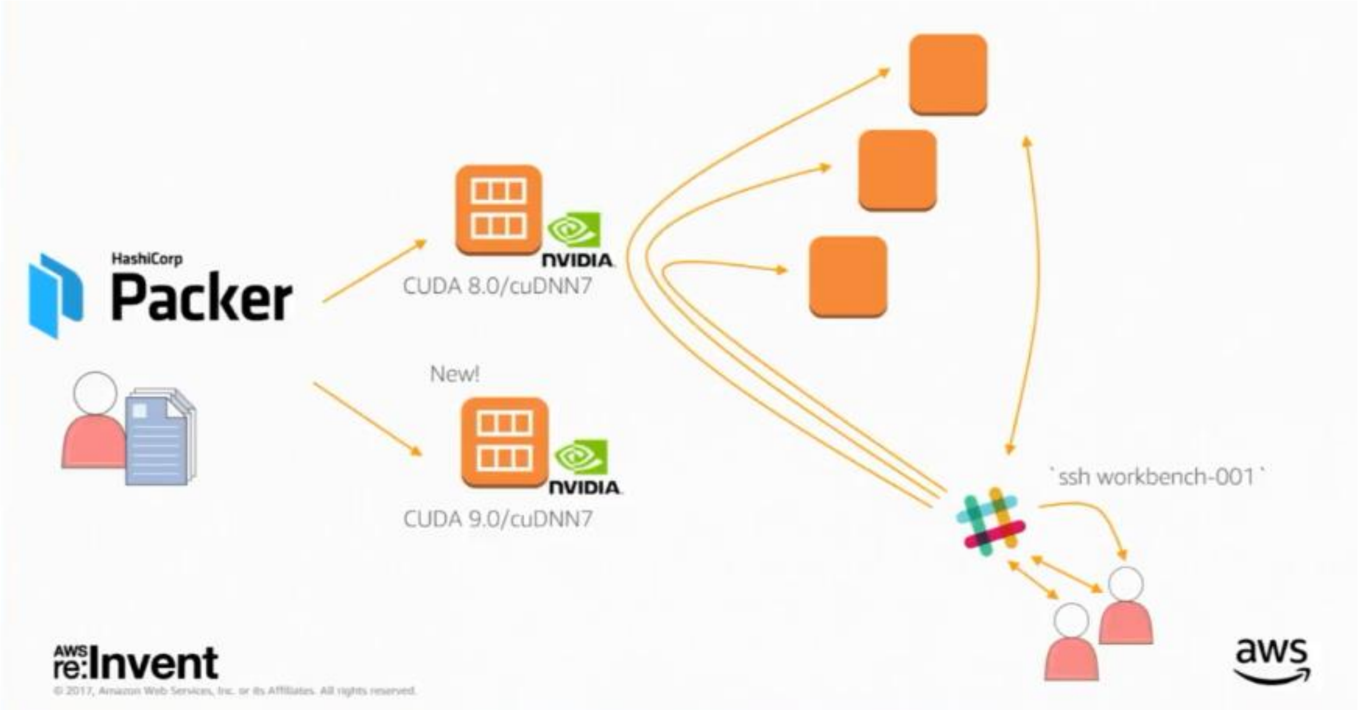
Rebuild dataset
making use of posterior insights


Labeled image dataset
Food Photos from Cookpad
Random Photos from other datasets (Nonfood)
Food/Nonfood image classifier

(Precision: 97.6%, Recall: 95.8%)

97.9% Accurate





Cooking log

- Scalable food/non-food image classification
  - Asynchronous and isolated architecture
  - Containerized GPU workloads
- `hako` makes it easy to define and deploy applications

Machine learning infrastructure

- Great environment makes our research fast and creative!
  - Managing multiple AMIs using Packer
  - Dedicated Instances
  - Operate instances via chat bot

We're hiring!

https://info.cookpad.com/us

https://github.com/cookpad

cookpad

AWS re:Invent

aws