



このコンテンツは公開から3年以上経過しており内容が古い可能性があります
最新情報については[サービス別資料](#)もしくはサービスのドキュメントをご確認ください

[AWS Black Belt Online Seminar]

Amazon SageMaker advanced

サービスカットシリーズ

Machine Learning Solutions Architect

宇都宮 聖子

2019/02/13

AWS 公式 Webinar

<https://amzn.to/JPWebinar>



過去資料

<https://amzn.to/JPArchive>



自己紹介

宇都宮 聖子

- 機械学習ソリューションアーキテクト
 - 機械学習サービスを担当
 - 前々職は量子情報の研究者
 - 前職は自動車OEMで自動運転開発
- 好きなサービス
 - Amazon SageMaker



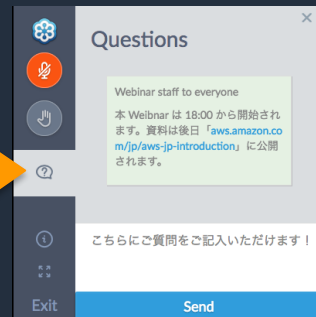
AWS Black Belt Online Seminar とは

「サービス別」「ソリューション別」「業種別」のそれぞれのテーマに分かれて、アマゾンウェブサービス ジャパン株式会社が主催するオンラインセミナーシリーズです。

質問を投げることができます！

- 書き込んだ質問は、主催者にしか見えません
- いただいたQ&Aをピックアップしてblogにご紹介させていただく場合がございます
- 今後のロードマップに関するご質問は
お答えできませんのでご了承下さい

- ① 吹き出しをクリック
- ② 質問を入力
- ③ Sendをクリック



Twitter ハッシュタグは以下をご利用ください
#awsblackbelt

内容についての注意点

- 本資料では2019年2月13日時点のサービス内容および価格についてご説明しています。最新の情報はAWS公式ウェブサイト(<http://aws.amazon.com>)にてご確認ください。
- 資料作成には十分注意しておりますが、資料内の価格とAWS公式ウェブサイト記載の価格に相違があった場合、AWS公式ウェブサイトの価格を優先とさせていただきます。
- 価格は税抜表記となっています。日本居住者のお客様が東京リージョンを使用する場合、別途消費税をご請求させていただきます。
- AWS does not offer binding price quotes. AWS pricing is publicly available and is subject to change in accordance with the AWS Customer Agreement available at <http://aws.amazon.com/agreement/>. Any pricing information included in this document is provided only as an estimate of usage charges for AWS services based on certain information that you have provided. Monthly charges will be based on your actual use of AWS services, and may vary from the estimates provided.

本日のアジェンダ

- 前提: Amazon SageMaker blackbelt の基本的な内容を習得されていること
- Amazon SageMaker の一歩進んだ使い方
 - データ準備: Amazon SageMaker Ground Truth
 - モデル開発: ビルトイン, 学習スクリプト, ML AWS Marketplace
 - 学習: 分散学習, Amazon SageMaker Neo
 - 推論: Elastic Inference
 - 強化学習
 - ワークフロー

Amazon SageMaker とは

- 機械学習システムでよくある問題を解消し、データサイエンティストやエンジニアが素早くプロセスを回せるようにするためのサービス
- 機械学習のインフラ構築・運用を自動化するだけでなく、そのほかのさまざまな機能も提供
- **東京リージョン**を含む、13 リージョンにてサービスを展開



ラベリング



開発



学習

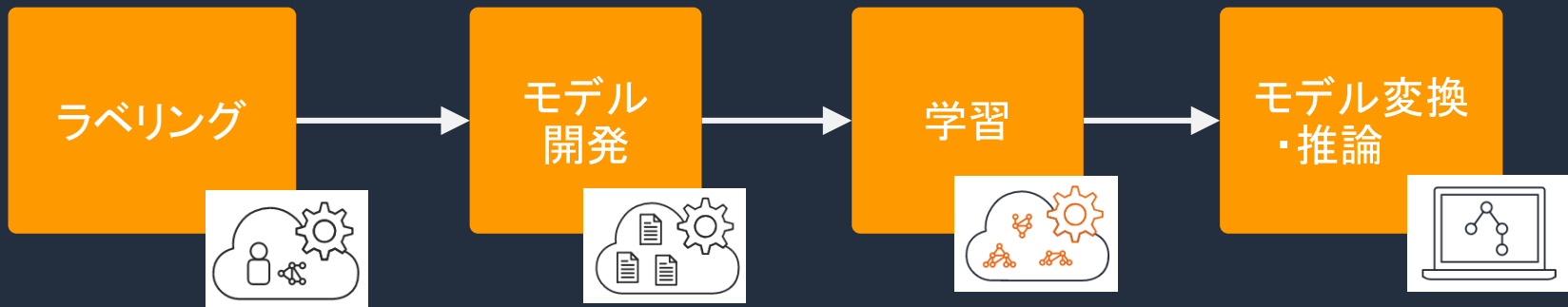


モデル変換



推論

SageMakerで行う機械学習の流れ



【ラベリング】 機械学習のための教師データラベリングの支援ツール

【開発】 学習するためのコードの記述や、入力データの加工整形を行うための環境

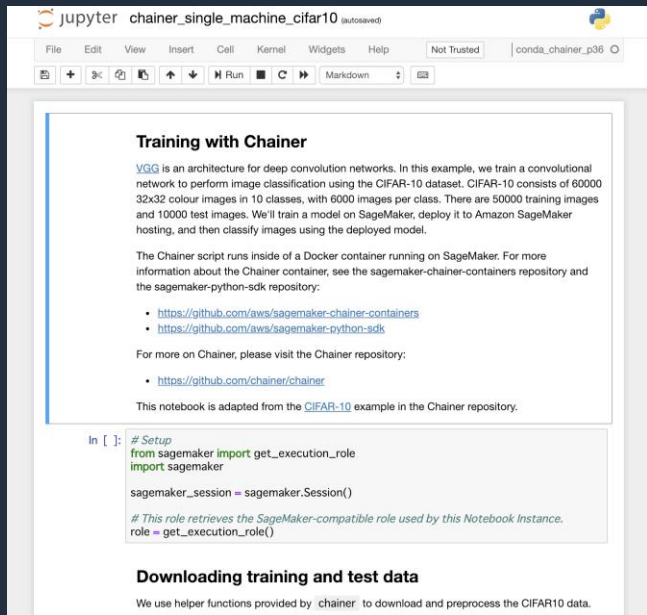
【学習】 API を叩くと学習用インスタンスが立ち上がり、学習ジョブを実行。複数ジョブの同時実行や、複数インスタンスでの分散学習、ハイパーパラメーターチューニング

【モデル変換・推論】 推論用のエンドポイントをデプロイ。推論におけるレイテンシを減らすため、学習済みモデルを実行環境に最適化された形へコンパイル

Amazon SageMaker Notebook instance

- SageMaker 上のワークフロー（環境・データのインポート、モデル定義、学習ジョブ、デプロイ、エンドポイント呼び出し）を SageMaker Python SDKで記述する

❖ Jupyter



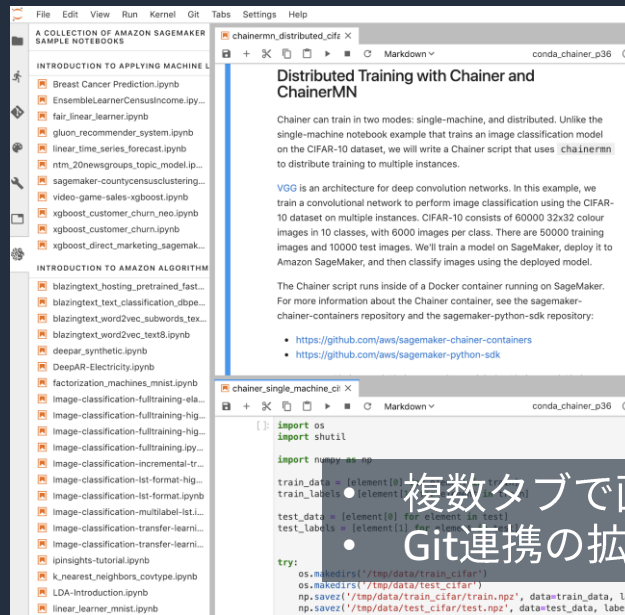
The screenshot shows a Jupyter Notebook interface with a document titled "Training with Chainer". The document contains text about VGG architecture, training on the CIFAR-10 dataset, and using Chainer for distributed training. It includes links to GitHub repositories for Chainer containers and the SageMaker Python SDK. At the bottom, there is a code cell with Python code for setting up the SageMaker environment.

```
In [ ]: # Setup
from sagemaker import get_execution_role
import sagemaker

sagemaker_session = sagemaker.Session()

# This role retrieves the SageMaker-compatible role used by this Notebook Instance.
role = get_execution_role()
```

❖ JupyterLab



The screenshot shows a JupyterLab interface with a collection of Amazon SageMaker sample notebooks. The interface is divided into three panes. The left pane shows a list of notebooks, including "Breast Cancer Prediction.ipynb", "EnsembleLearnerCensusincome.ipynb", "fair_linear_learner.ipynb", "gluon_recommender_system.ipynb", "linear_time_series_forecast.ipynb", "rtm_20newsgroups_topic_model.ipynb", "sagemaker-countycensusclustering.ipynb", "video-game-sales_xgboost.ipynb", "xgboost_customer_churn_neo.ipynb", "xgboost_customer_churn.ipynb", and "xgboost_direct_marketing_sagemaker.ipynb". The middle pane shows the "Introduction to Applying Machine Learning" section of a notebook. The right pane shows the "Distributed Training with Chainer and ChainerMN" section, which includes text about training a convolutional network on the CIFAR-10 dataset and a code cell for setting up the SageMaker environment.

```
[ ]: import os
import shutil

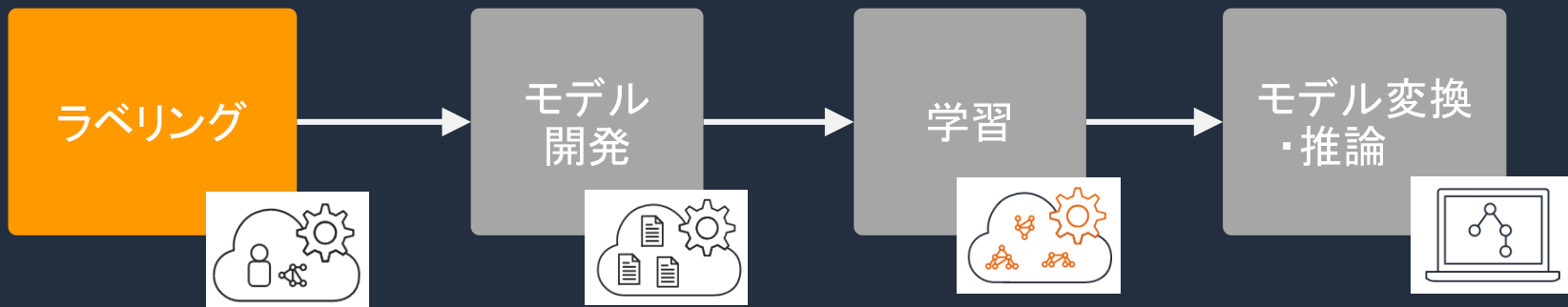
import numpy as np

train_data = [element[0] for element in train_labels]
test_data = [element[0] for element in test_labels]

try:
    os.makedirs('/tmp/data/train_cifar')
    os.makedirs('/tmp/data/test_cifar')
    np.savez('/tmp/data/train_cifar/train.npz', data=train_data, label=train_labels)
    np.savez('/tmp/data/test_cifar/test.npz', data=test_data, label=test_labels)
```

複数タブで画面管理
• Git連携の拡張機能

アノテーション



データへのラベル付け(アノテーション)にはコスト・時間がかかる

- 効率の良いアノテーションツールの作成
- 作業を割り当てるワーカーの募集
- 進捗管理・作業割り振り
- 大量データのラベル付け

Amazon SageMaker Ground Truth

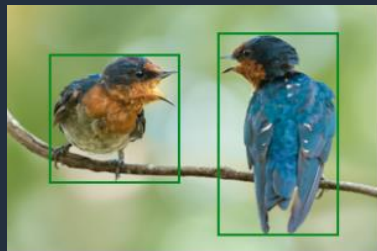
データにラベル (Ground Truth) を付与するアノテーション作業の支援サービス

- アノテーションにおける一般的なワークフローの管理ツール
- ラベルを付与するワーカーは, **Amazon Mechanical Turk**, **外部ベンダ(AWS Marketplace)**, **自社のプライベートチーム** の3つから選択
- 以下の4種類のタスク向けアノテーションツールも提供(カスタムも可能)

画像分類



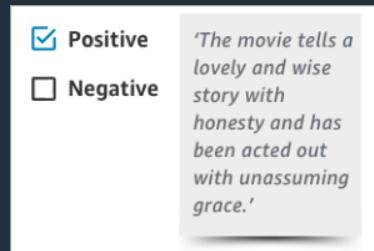
物体検出



セマンティック セグメンテーション



文章分類



組み込みのアノテーションツール紹介

画像分類

アノテーションの基本プロセス



1. アノテーション対象のデータをアップロード



Amazon S3

3. タスクはワーカーに自動で割り振られる



ワーカー



SageMaker ユーザ

2. ラベリングジョブの作成



Amazon SageMaker

5. アノテーション結果がS3集計される

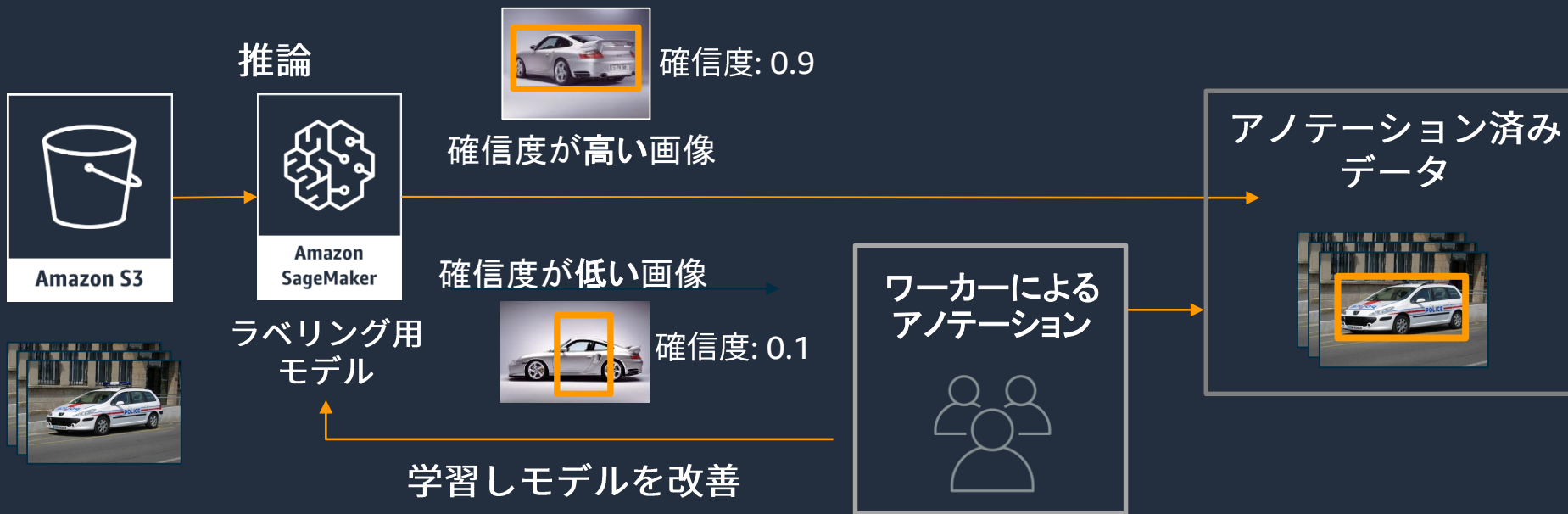
4. アノテーションツールでワーカーがアノテーションを行う



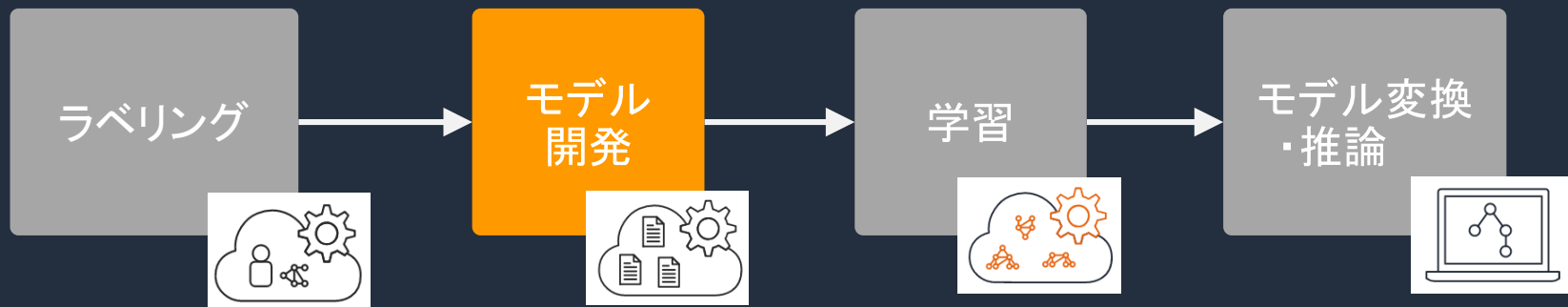
複数人の結果をマージできる

自動ラベリング (オプション機能)

大規模データセット (5000データ以上) にラベリングする際、数割を
ワーカーでラベル付けし、残りを自動化することで、時間とコストを削減



機械学習のモデルを開発する



モデル開発

- モデル生成の方法
 - ビルトインアルゴリズムを使う
 - SageMaker コンテナ対応のフレームワークを使ったモデル開発
 - AWS Marketplace Machine Learning でモデル購入

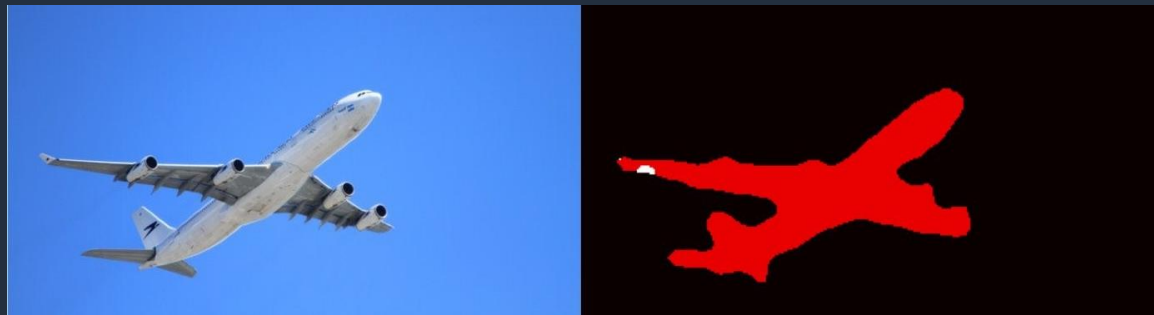
モデル開発の選択肢

- Amazon SageMaker **ビルトインアルゴリズム**を使う
- 自前の学習用スクリプトを使う
- **AWS Marketplace** で機械学習モデルを購入する

SageMaker ビルトインアルゴリズムを使う

- SageMaker上での実装を最適化した、機械学習アルゴリズム
- アルゴリズムごとにコンテナが準備されており、分散学習などが簡単に使える

Semantic segmentation: ピクセル単位のラベリング



- [Fully-Convolutional Network \(FCN\) algorithm](#), [Pyramid Scene Parsing \(PSP\) algorithm](#), [DeepLabV3](#) のいずれかからアルゴリズムを選択

SageMaker ビルトインアルゴリズム

～ 機械学習モデル ～

モデル名	教師データ	アルゴリズム説明	利用用途の例
Linear Learner	あり	線形回帰	分類・回帰などの分析
XGBoost	あり	XGBoost, 勾配ブーストツリー (eXtreme Gradient Boosting)	分類・回帰などの分析
PCA	なし	主成分分析 (Principal Component Analysis)	次元削減
k-means	なし	K平均法	クラスタリング
k-NN	あり	K近傍法	クラスタリング
Factorization Machines	あり	行列分解	レコメンド, 回帰, 分類
Random Cut Forest	なし	robust random cut tree	時系列データの異常検知
LDA (Latent Dirichlet Allocation)	あり	生成的統計モデル	トピックモデル

※ LDAのオリジナルは教師なし

<https://docs.aws.amazon.com/sagemaker/latest/dg/algos.html>

SageMaker ビルトインアルゴリズム ~ ディープラーニング モデル ~

	モデル名	教師データ	アルゴリズム	利用用途の例
画像処理	Image classification	あり	ResNet	画像の多値分類
	Object Detection	あり	SSD (Single Shot multibox Detector)	物体の画像内領域をバウンディングボックスで検出
	Semantic Segmentation	あり	FCN, PSP, DeepLabV3 (ResNet50, ResNet101)	ピクセル単位の画像内の物体領域検出
自然言語処理	seq2seq	あり	Deep LSTM	テキスト要約, 音声認識
	Neural Topic Model	なし	NTM, LDA	テキストデータの構造化
	Blazing text	なし	Word2Vec	センチメント分析
		あり	Text Classification	単語のマイニング
	Object2Vec	あり	Word2Vec 一般ベクトル化	分類, レコメンド
時系列	DeepAR Forecasting	あり	Autoregressive RNN	確率的な時系列予測
異常検知	IP Insights	なし	NN (IPとentityの関連付け)	悪意あるIPアドレスの検出

自前の学習用スクリプトを使う

- SageMaker でサポートしているフレームワーク一覧 ※ 2019年2月13日時点の情報です

	フレームワーク	SageMaker container サポートバージョン
Deep learning	TensorFlow	Legacy mode: 1.4.1, 1.5.0, 1.6.0, 1.7.0, 1.8.0, 1.9.0, 1.10.0 Script mode: 1.11.0, 1.12.0
	Chainer	4.0.0, 4.1.0, 5.0.0
	PyTorch	0.4.0, 1.0.0
	MXNet	1.3.0, 1.2.1, 1.1.0, 0.12.1
ML	scikit-learn	0.20.0

<https://github.com/aws/sagemaker-python-sdk/tree/master/src/sagemaker>

TensorFlow: <https://github.com/aws/sagemaker-python-sdk/tree/master/src/sagemaker/tensorflow>

Chainer: <https://github.com/aws/sagemaker-python-sdk/tree/master/src/sagemaker/chainer>

PyTorch: <https://github.com/aws/sagemaker-python-sdk/tree/master/src/sagemaker/pytorch>

MXNet: <https://github.com/aws/sagemaker-python-sdk/tree/master/src/sagemaker/mxnet>

Sklearn: <https://github.com/aws/sagemaker-python-sdk/tree/master/src/sagemaker/sklearn>

TensorFlowとMXNet が Script Modeに対応

- TensorFlow
 - Ver.1.11以降はscript mode対応, Legacy modeサポートは1.12まで
 - これまでpython2系のみだったが, **ver.1.11からpython3系もサポート**
 - Elastic Inferenceは1.11.0, 1.12.0 のみ
- MXNet
 - Ver. 1.3.0, 1.2.1, 1.1.0, 0.12.1
- PyTorch, Chainerは元々Script mode

<https://github.com/aws/sagemaker-python-sdk/blob/master/src/sagemaker/tensorflow/README.rst>

<https://github.com/aws/sagemaker-python-sdk/blob/master/src/sagemaker/mxnet/README.rst>



TensorFlow,
MXNet code

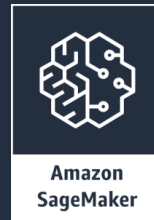


Script mode 対応
Container



`if __name__=="__main__"`

に学習スクリプトを書く



TensorFlow が Script Mode に対応

Script Mode では、スクリプトの変更が少なく済むように

```
import argparse
import os

if __name__ == '__main__':

    parser = argparse.ArgumentParser()

    # hyperparameters sent by the client are passed as command-line arguments to the script.
    parser.add_argument('--epochs', type=int, default=10)
    parser.add_argument('--batch_size', type=int, default=100)
    parser.add_argument('--learning_rate', type=float, default=0.1)

    # input data and model directories
    parser.add_argument('--model_dir', type=str)
    parser.add_argument('--train', type=str, default=os.environ.get('SM_CHANNEL_TRAIN'))
    parser.add_argument('--test', type=str, default=os.environ.get('SM_CHANNEL_TEST'))

    args, _ = parser.parse_known_args()

    # ... load from args.train and args.test, train a model, write model to args.model_dir.
```

クライアントから送信されたハイパーパラメータは、
コマンドライン引数としてスクリプトに渡される

入力データとモデルのディレクトリ

main ガードの中に学習処理を
そのまま書く

SageMaker 側で引数として渡
される hyperparameter や入
力データのパスなどを
argparse でパースして読み取る

<https://github.com/aws/sagemaker-python-sdk/blob/master/src/sagemaker/tensorflow/README.rst>

AWS Marketplace から機械学習のモデルを購入する

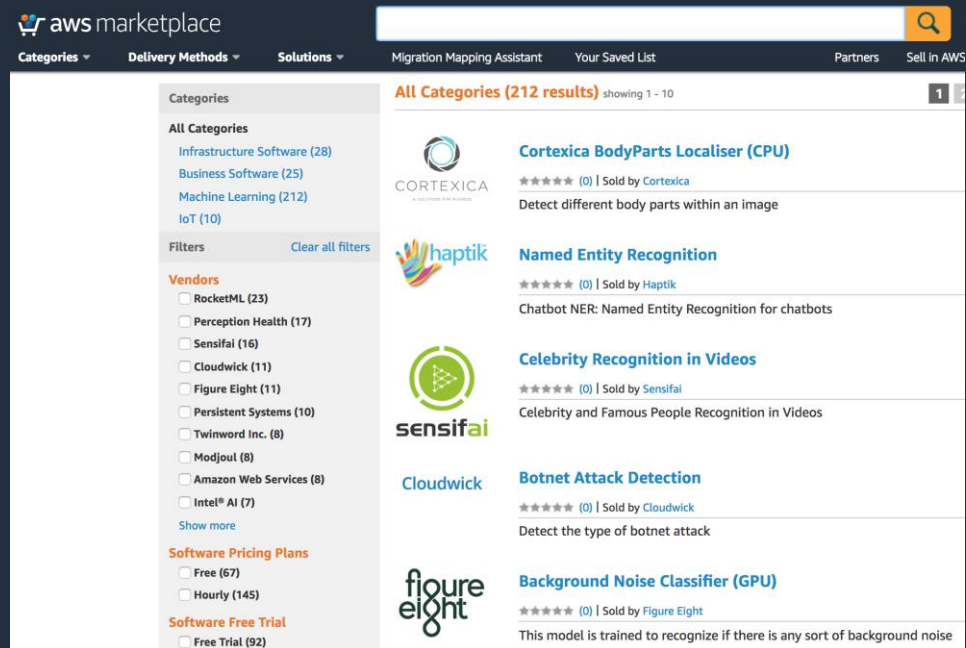
- AWSマーケットプレイス経由で、SageMaker上で使う機械学習モデルの売買が可能。小売、メディア向けなど200以上のアルゴリズムがすでに公開済み

アルゴリズム購入者：

Amazon SageMaker で学習ジョブおよび、推論エンドポイント（バッチ推論ジョブもok）

アルゴリズム購買者：

モデルの中身を秘匿してモデルの出品が可能



The screenshot displays the AWS Marketplace interface. The top navigation bar includes 'aws marketplace', 'Categories', 'Delivery Methods', 'Solutions', and a search bar. Below the navigation, there are sections for 'Categories' (All Categories, Infrastructure Software (28), Business Software (25), Machine Learning (212), IoT (10)), 'Filters' (Clear all filters), 'Vendors' (RocketML (23), Perception Health (17), Sensifai (16), Cloudwick (11), Figure Eight (11), Persistent Systems (10), Twinword Inc. (8), Modjoul (8), Amazon Web Services (8), Intel® AI (7), Show more), 'Software Pricing Plans' (Free (67), Hourly (145)), and 'Software Free Trial' (Free Trial (92)). The main content area shows a list of models under the heading 'All Categories (212 results) showing 1 - 10'. The first few models listed are: 'Cortexica BodyParts Localiser (CPU)' (5 stars, 0 reviews, sold by Cortexica), 'Named Entity Recognition' (5 stars, 0 reviews, sold by Haptik), 'Celebrity Recognition in Videos' (5 stars, 0 reviews, sold by Sensifai), 'Botnet Attack Detection' (5 stars, 0 reviews, sold by Cloudwick), and 'Background Noise Classifier (GPU)' (5 stars, 0 reviews, sold by Figure Eight).

MLマーケットプレイスアルゴリズムの利用(学習)

Marketplace上で
アルゴリズムの選択

The screenshot shows the AWS Marketplace listing for 'Text Similarity Analyzer' by TIBCO Software Inc. The listing includes a 'Continue to Subscribe' button, a 'Save to List' button, and a navigation menu with 'Overview', 'Pricing', 'Usage', 'Support', and 'Reviews'. The 'Overview' section is active, showing the product name and a brief description: 'This algorithm produces similarity scores for a document or a line of text'.

SageMaker上で
アルゴリズム登録

The screenshot shows the SageMaker console interface for the 'Text Similarity Analyzer' algorithm. It features buttons for 'チューニングジョブの作成' (Create Tuning Job) and 'トレーニングジョブの作成' (Create Training Job). Below these is a table titled 'アルゴリズムのバージョン' (Algorithm Versions) with the following data:

タイトル	バージョン	アルゴリズム ARN
Text Similarity Analyzer	v1	arn:aws:sagemaker:us-west-2:594846645681:algorithm/nlp-text-similarity-1543457191-707249cf62201cf1016455c156894b9e

トレーニングジョブ
の作成・実行

The screenshot shows the 'トレーニングジョブの作成' (Create Training Job) section in the SageMaker console. It contains a paragraph of text explaining the process: 'トレーニングジョブを実行すると、Amazon SageMaker は分散コンピューティングクラスターをセットアップし、トレーニングを実行して、トレーニングが完了するとクラスターを削除します。結果として生成されるモデルアーティファクトが、トレーニングジョブの作成時に指定した場所に保存されます。' followed by a link for more details. Below the text is a section titled 'ジョブの設定' (Job Configuration).

MLマーケットプレイスモデルの利用(推論)

Marketplace 上で
モデルの選択

The screenshot shows the product page for 'Deep Vision vehicle recognition' on AWS Marketplace. It includes a 'Free Trial' badge, a 'Continue to Subscribe' button, and a 'Save to List' button. The product description states: 'Recognize vehicles with Deep Vision visual recognition API. Get car detections, the year, make and model, and car views from any angle.' The page has tabs for Overview, Pricing, Usage, Support, and Reviews. The 'Product Overview' section describes the API's capabilities and includes a 'Highlights' box with the following text: 'AI-powered visual recognition API for your product, services, and applications. Integrate into AWS workflow.'

SageMaker 上での
モデルパッケージ登録

The screenshot shows the 'Model packages' section in the AWS SageMaker console. It features a search bar, 'Find model packages' and 'Manage subscriptions' buttons, and a table with the following data:

Product title	Version
Deep Vision vehicle recognition	1.0

エンドポイントの作成

The screenshot shows the 'Endpoints' section in the AWS SageMaker console. It includes a search bar, 'Update endpoint' and 'Create endpoint' buttons, and a table with the following data:

Name	ARN	Creation time	Status	Last updated
DV-vehicle-detection-endpoint-2018-11-27-08-28-49	arn:aws:sagemaker:us-east-2:123456789012:endpoint/dv-vehicle-detection-endpoint-	Nov 27, 2018 08:29 UTC	InService	Nov 27, 2018 08:36 UTC

MLマーケットプレイス 価格

SageMakerコンソール, SDK,
AWS CLIからアクセス可能

価格：モデルパッケージの利用に対し、
ソフトウェア使用料と AWS のリソース
使用料を利用時間に応じて課金

- アルゴリズム学習
- モデルリアルタイム推論
- モデルバッチ推論

Pricing Information

Use this tool to estimate the software and infrastructure costs based your configuration choices. Your usage and costs might be different from this estimate. They will be reflected on your monthly AWS billing reports.

Estimating your costs

Choose your region and launch option to see the pricing details. Then, modify the estimated price by choosing different instance types.

Region

US East (N. Virginia)

Fulfillment Option

Amazon SageMaker

Software Pricing

Algorithm Training **\$1/hr**
running on ml.p3.8xlarge ▶

Model Realtime Inference **\$2.6/hr**
running on ml.p3.2xlarge ▶

Model Batch Transform **\$2.6/hr**
running on ml.p3.8xlarge ▶

Infrastructure Pricing

SageMaker Algorithm Training **\$17.136/host/hr** ⓘ
running on ml.p3.8xlarge

SageMaker Realtime Inference **\$4.284/host/hr** ⓘ
running on ml.p3.2xlarge

SageMaker Batch Transform **\$17.136/host/hr** ⓘ
running on ml.p3.8xlarge

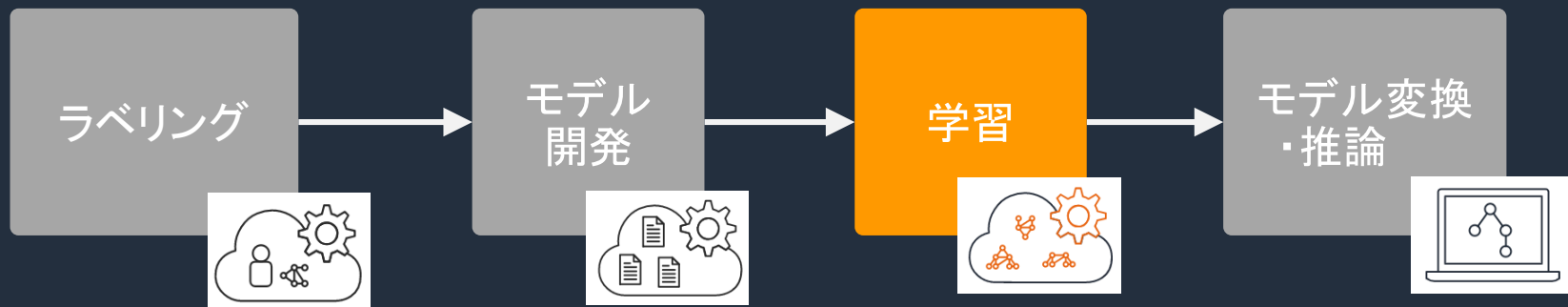
Algorithm Training

The table shows current software and infrastructure pricing for services hosted in US East (N. Virginia). Additional taxes or fees may apply.

	InstanceType	Algorithm/hr	SageMaker/hr	Total/hr
<input type="radio"/>	ml.p2.xlarge	\$ 1	\$ 1.26	\$ 1.36
<input type="radio"/>	ml.p2.8xlarge	\$ 7	\$ 10.08	\$ 10.78
<input type="radio"/>	ml.p2.16xlarge	\$ 1	\$ 20.16	\$ 21.16
<input type="radio"/>	ml.p3.2xlarge	\$ 4	\$ 4.284	\$ 4.684
<input checked="" type="radio"/>	ml.p3.8xlarge <i>Vendor Recommended</i>	\$ 1	\$ 17.136	\$ 18.136
<input type="radio"/>	ml.p3.16xlarge	\$ 2	\$ 34.272	\$ 36.272

学習・推論モデルは
インスタンスごとに
価格が設定される

機械学習のモデルを開発する



- TensorFlow (Horovodにも対応), MXNet, PyTorch, Chainer それぞれのフレームワークに適した分散学習を提供. training instance 数の指定ですぐに分散学習が利用できる
- ハイパーパラメータ最適化

TensorFlow で分散学習をする場合

- Parameter server を使った分散学習

```
from sagemaker.tensorflow import TensorFlow

tf_estimator = TensorFlow(entry_point='tf-train.py', role='SageMakerRole',
                          train_instance_count=2, train_instance_type='ml.p2.xlarge',
                          framework_version='1.11', py_version='py3',
                          distributions={'parameter_server': {'enabled': True}})
tf_estimator.fit('s3://bucket/path/to/training/data')
```

- Horovod を使った分散学習 TF v1.12以降で利用可能

```
from sagemaker.tensorflow import TensorFlow

tf_estimator = TensorFlow(entry_point='tf-train.py', role='SageMakerRole',
                          train_instance_count=1, train_instance_type='ml.p2.xlarge',
                          framework_version='1.12', py_version='py3',
                          distributions={
                              'mpi': {
                                  'enabled': True,
                                  'processes_per_host': 2,
                                  'custom_mpi_options': '--NCCL_DEBUG INFO'
                              }
                          })
tf_estimator.fit('s3://bucket/path/to/training/data')
```

`mpi`: trueにすると mpirun コマンドが使えるようになる

<https://github.com/aws/sagemaker-python-sdk/blob/master/src/sagemaker/tensorflow/README.rst>



ChainerMN を使った分散学習

- [ChainerMN](#) によるMPI(Message Passing Interface)を用いた分散学習
- 2019年2月12日時点では、ChainerMN の対応は ver. Chainer 4.1 まで
- フレームワークバージョンが指定されない場合は、デフォルトで 4.1.0 が選択される
- `train_instance_count` が 2 より大きい場合に、`mpirun` で学習スクリプトを実行

```
from sagemaker.chainer.estimator import Chainer

chainer_estimator = Chainer(entry_point='chainer_cifar_vgg_distributed.py',
                             source_dir="src",
                             role=role,
                             sagemaker_session=sagemaker_session,
                             use_mpi=True,
                             train_instance_count=2,
                             train_instance_type='ml.p2.xlarge',
                             hyperparameters={'epochs': 30, 'batch-size': 256})
```

```
chainer_estimator.fit({'train': train_input, 'test': test_input})
```

```
WARNING:sagemaker:No framework_version specified, defaulting to version 4.1.0. This is not the latest supported version. If you would like to use version 5.0.0, please add framework_version=5.0.0 to your constructor.
INFO:sagemaker:Creating training-job with name: sagemaker-chainer-2019-01-20-05-30-01-031
```

<https://github.com/aws/sagemaker-python-sdk/blob/master/src/sagemaker/tensorflow/README.rst>

ハイパーパラメータ最適化 (HPO) の書き方

1. Chainer estimator 初期化の際に hyperparameter を指定
 2. HyperparameterTuner に hyperparameter_ranges を渡す。連続・離散の範囲指定ジョブ内で **ベイズ最適化によるHPO**
 3. HyperparameterTuner にチューニング対象の (2) や、ターゲットメトリクスを指定、CloudWatch Logs に抽出される
- HPO ウォームスタート機能も利用可能

```
from sagemaker.chainer import Chainer
```

```
estimator = Chainer(entry_point="mnist.py",  
                    role=role,  
                    framework_version='5.0.0',  
                    train_instance_count=1,  
                    train_instance_type='ml.m4.xlarge',  
                    ① hyperparameters={'epochs': 30, 'batch-size': 256})
```

```
hyperparameter_ranges = {'lr': ContinuousParameter(0.001, 0.1),  
                          ② 'batch-size': CategoricalParameter([32,64,128,256,512])}
```

```
tuner = HyperparameterTuner(estimator,  
                             objective_metric_name,  
                             hyperparameter_ranges,  
                             metric_definitions,  
                             ③ max_jobs=9,  
                             max_parallel_jobs=3,  
                             objective_type=objective_type)
```

```
tuner.fit({'training': inputs})
```

<https://github.com/aws/sagemaker-python-sdk#sagemaker-automatic-model-tuning>

[https://aws.amazon.com/jp/blogs/news/amazon-sagemaker-automatic-model-tuning-becomes-](https://aws.amazon.com/jp/blogs/news/amazon-sagemaker-automatic-model-tuning-becomes-more-efficient-with-warm-start-of-hyperparameter-tuning-jobs/)

[more-efficient-with-warm-start-of-hyperparameter-tuning-jobs/](https://aws.amazon.com/jp/blogs/news/amazon-sagemaker-automatic-model-tuning-becomes-more-efficient-with-warm-start-of-hyperparameter-tuning-jobs/)



ハイパーパラメータ探索の結果をコンソールで確認

目標メトリクス値に基づいて探索した結果を

The screenshot displays the AWS SageMaker console interface for hyperparameter tuning. The main view shows the 'Best Training Job Summary' for a specific training job, highlighting the 'average test loss' metric. A secondary window shows a list of training jobs, with the 'average test loss' column highlighted.

最善のトレーニングジョブの概要

This training job is the best training job for only this hyperparameter tuning job. [モデルの作成](#)

名前	ステータス	目標メトリクス	値
sagemaker-pytorch-181202-1543-006-af44e76f	Completed	average test loss	0.04439999908208847

トレーニングジョブの最善のハイパーパラメータ

名前	タイプ	値
_tuning_objective_metric	FreeText	average test loss
backend	FreeText	"gloo"
batch-size	Categorical	"64"
epochs	FreeText	6
lr	Continuous	0.06837402575909636
sagemaker_container_log_level	FreeText	20
sagemaker_enable_cloudwatch_metrics	FreeText	false
sagemaker_estimator_class_name	FreeText	"PyTorch"
sagemaker_estimator_module	FreeText	"sagemaker.pytorch.estimator"

最善のトレーニングジョブ

トレーニングジョブのステータスカウンター

完了済み 9 進行中 0 停止 0 失敗 0 (再試行可能: 0, 再試行不可能: 0)

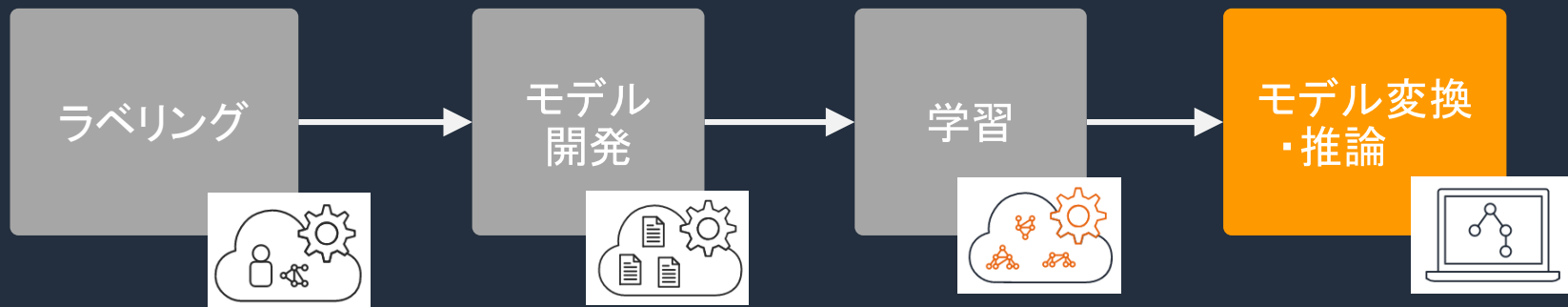
トレーニングジョブ

目標メトリクス値によるソートでは、メトリクス値があるジョブのみが表示されます。

名前	ステータス	目標メトリクス値
sagemaker-pytorch-181202-1543-009-6c755737	Completed	0.1006999984383583
sagemaker-pytorch-181202-1543-008-3305251a	Completed	0.1136000007390976
sagemaker-pytorch-181202-1543-007-151349f0	Completed	0.0681999996304512
sagemaker-pytorch-181202-1543-006-af44e76f	Completed	0.04439999908208847
sagemaker-pytorch-181202-1543-005-79fa8647	Completed	0.05290000140666962



学習済みモデルの変換・推論



モデル変換・推論

- SageMaker Neo: モデル変換による推論の高速化
- Elastic Inference: GPUアクセラレータ追加による推論のコスト削減と高速化

Amazon SageMaker Neo

- SageMaker で学習したモデルを，推論環境に最適化された形にコンパイル
- デプロイメントプラットフォーム上のリソース使用料を1/10程度に削減，推論の高速化
- MLフレームワーク固有の機能を，どこでも実行できる単一のコンパイル済み環境に変換
- EC2 インスタンスや Greengrass デバイス上で高速に動作するように変換する



<https://aws.amazon.com/sagemaker/neo/>

SageMaker Neo によるモデルのコンパイル

- 各種MLフレームワークやビルトインアルゴリズムに対応
- 対応リージョンはバージニア北部, オレゴン, オハイオ, アイルランド
- Neoの利用料金は無料 (インスタンスでのジョブ実行料金のみ)

対応フレームワーク	対応プラットフォーム
<ul style="list-style-type: none">• TensorFlow• MXNet• PyTorch• ONNX• XGBoost	<ul style="list-style-type: none">• EC2 インスタンス (ml.c4, ml.c5, ml.m4, ml.m5, ml.p2, ml.p3)• Jetson TX1/2• DeepLens• Raspberry Pi 3 Model

<https://aws.amazon.com/jp/blogs/news/amazon-sagemaker-neo-train-your-machine-learning-models-once-run-them-anywhere/>

<https://docs.aws.amazon.com/sagemaker/latest/dg/neo.html>

SageMaker Neo の Python SDK による利用の流れ

- TensorFlowEstimator.compile_model メソッドで学習モデルを、ターゲットデバイスに最適なコンパイル

学習

```
mnist_estimator = TensorFlow(  
    entry_point='mnist.py', role=role, framework_version='1.11.0',  
    training_steps=1000, evaluation_steps=100,  
    train_instance_count=2, train_instance_type='ml.c4.xlarge')
```

```
mnist_estimator.fit(inputs)
```

C5向けに
最適化

```
optimized_estimator = mnist_estimator.compile_model(  
    target_instance_family='ml_c5', input_shape={'data':[1, 784]},  
    output_path=output_path,  
    framework='tensorflow', framework_version='1.11.0')
```

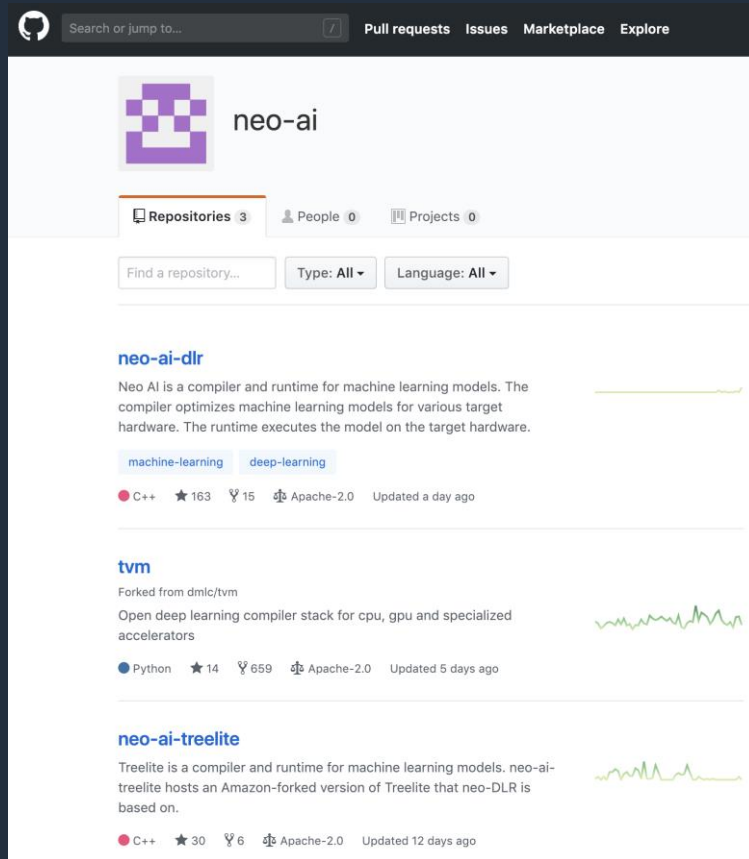
C5にデプロイ

```
optimized_predictor = optimized_estimator.deploy(  
    initial_instance_count = 1, instance_type = 'ml.c5.4xlarge')
```

https://github.com/aws-labs/amazon-sagemaker-examples/blob/master/sagemaker-python-sdk/tensorflow_distributed_mnist/tensorflow_distributed_mnist_neo.ipynb

OSS Neo-AI プロジェクト

- Apache Software License に基づく [Neo-AI プロジェクト](#)を2019年1月発表
- ワシントン大学の OSS 研究プロジェクト DMLC ([Distributed Machine Learning Common](#)) のTVMとTreeliteを拡張
 - TVM: Deep learning stack compiler for CPU, GPU.
 - Treelite: Decision tree compiler
- LLVMや[Halide](#)などの従来のコンパイラテクノロジーに関する数十年にわたる研究に基づいて構築されたランタイム
- 現在は AWS, ARM, Intel, NVIDIAをサポート
- Cadence, Qualcomm, Xilinx にも対応予定



The screenshot shows the GitHub profile for the 'neo-ai' organization. At the top, there are navigation links for 'Pull requests', 'Issues', 'Marketplace', and 'Explore'. The profile header includes the organization's logo (a purple and white checkerboard pattern) and the name 'neo-ai'. Below this, there are statistics for 'Repositories' (3), 'People' (0), and 'Projects' (0). A search bar and filters for 'Type' and 'Language' are present. The repository list includes:

- neo-ai-dlr**: Neo AI is a compiler and runtime for machine learning models. The compiler optimizes machine learning models for various target hardware. The runtime executes the model on the target hardware. It is a C++ project with 163 stars and 15 forks, using Apache-2.0 license, updated a day ago. It is categorized under 'machine-learning' and 'deep-learning'.
- tvm**: Forked from dmlc/tvm. Open deep learning compiler stack for cpu, gpu and specialized accelerators. It is a Python project with 14 stars and 659 forks, using Apache-2.0 license, updated 5 days ago.
- neo-ai-treelite**: Treelite is a compiler and runtime for machine learning models. neo-ai-treelite hosts an Amazon-forked version of Treelite that neo-DLR is based on. It is a C++ project with 30 stars and 6 forks, using Apache-2.0 license, updated 12 days ago.

<https://github.com/neo-ai/>



CPU インスタンスの計算を GPU でアクセラレート

- スタンドアロンのGPU は主に学習に最適化されており、推論には大きすぎる
- リアルタイムの推論では 少量のGPUしか消費せず高コスト
- 適切なGPU リソースをCPUに関連づけることで、**高速な推論を低コストで実行**
- 画像分類の推論処理をする場合、コストを下げながら十分なパフォーマンスを得ることが可能



インスタンス	EIA	推論速度 (msec)	価格 (\$/hour)
c5.large	なし	230 msec	\$0.085
c5.large	eia1.medium	46 msec	\$0.22
p2.xlarge	なし	42 msec	\$0.90

価格は 2018/11 時点のバージニア北部のもの

<https://aws.amazon.com/jp/blogs/news/amazon-elastic-inference-gpu-powered-deep-learning-inference-acceleration/>

Amazon Elastic Inference

- 推論に適した低コストのGPU駆動のアクセラレーションを，CPU の EC2 および SageMaker インスタンスに適用．DL 実行コストを最大75%削減
- 最大 32TFLOPSの混合精度演算を提供できる
- 利用方法
 - エンドポイントのインスタンスにEIAアタッチして利用
 - ローカルモード推論時は SageMaker Notebook インスタンスにEIAをアタッチ

Accelerator type	TFLOPS FP32 throughput	TFLOPS FP16 throughput	Memory in GB
ml.eia1.medium	1	8	1
ml.eia1.large	2	16	2
ml.eia1.xlarge	4	32	4

<https://docs.aws.amazon.com/sagemaker/latest/dg/ei.html>

Amazon Elastic Inference の利用

- SageMaker Python SDK の TensorFlow または MXNet と、SageMaker 用コンテナを使用
- 他のフレームワークは ONNX を使ってエクスポートし、MXNet にインポートして利用
- Image Classification, Object Detection のいずれかの組み込みアルゴリズムを使用
- TensorFlow または MXNet の EI バージョン対応の独自コンテナを構築
- 例) TensorFlow の場合:
 - Estimator または Model オブジェクトの `deploy` メソッドで EIA を指定

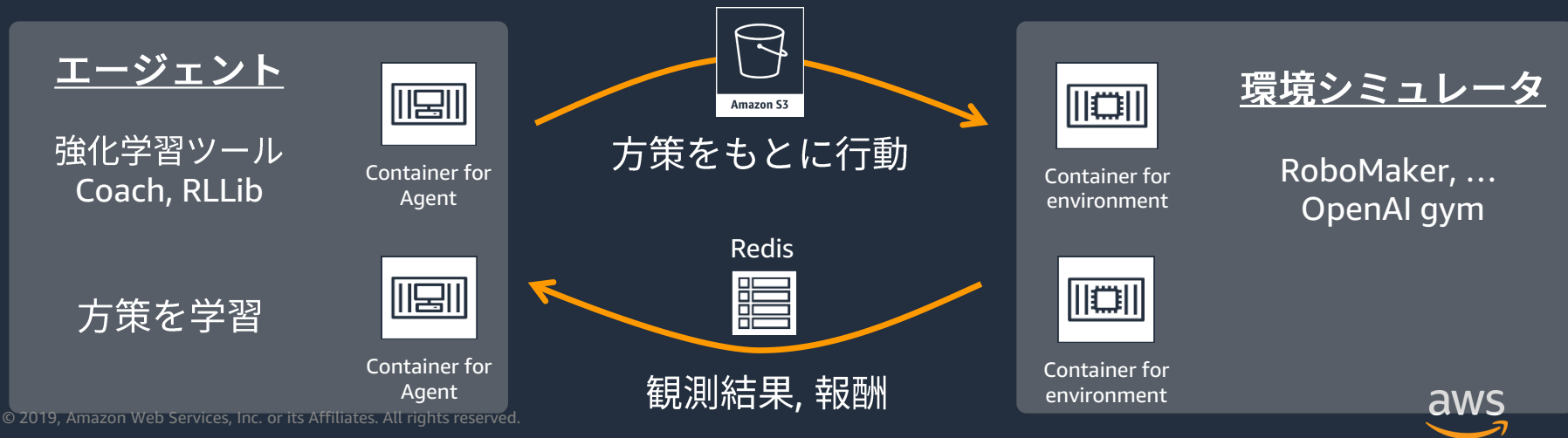
```
# Deploy an estimator using EI (using the accelerator_type input argument)  
predictor = estimator.deploy(initial_instance_count=1,  
                             instance_type='ml.m4.xlarge',  
                             accelerator_type='ml.eia1.medium')
```

```
# Deploy a model using EI (using the accelerator_type input argument)  
predictor = model.deploy(initial_instance_count=1,  
                          instance_type='ml.m4.xlarge',  
                          accelerator_type='ml.eia1.medium')
```

https://docs.aws.amazon.com/ja_jp/sagemaker/latest/dg/ei.html

Amazon SageMaker Reinforcement Learning

- ゲームやロボットのシミュレーション環境と統合した SageMaker 上の強化学習
- 強化学習ツールキットとして Coach と RL-Ray をサポート
- AWS RoboMaker, Roboschoolなど OSS の OpenAI Gym interface 経由で利用可能
- 強化学習ライブラリを利用し，分散学習環境（学習の分散と環境の分散が可能）



SageMaker 強化学習の構成

~ RoboMakerを使ったDeepRacer~

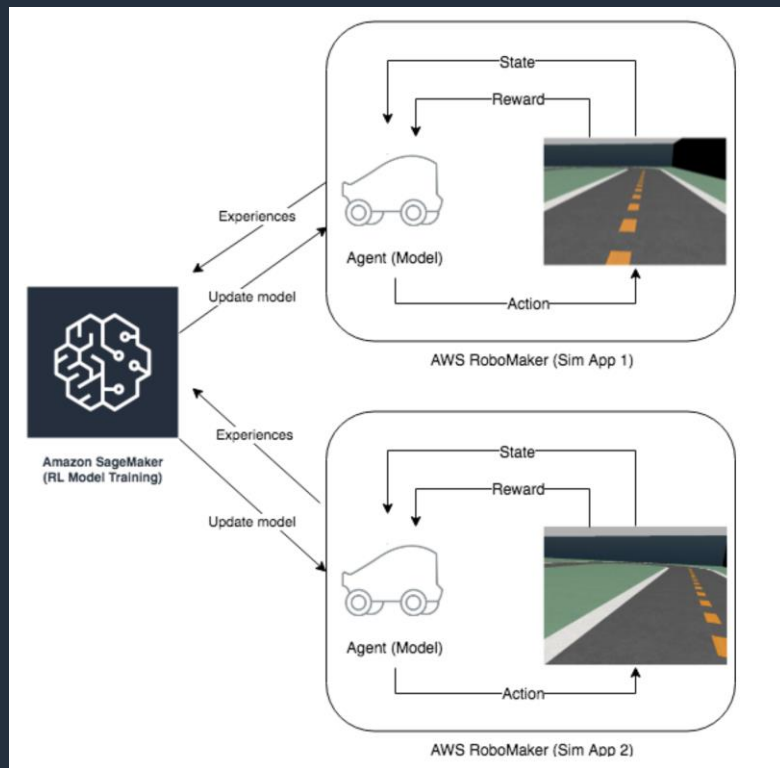
1. 目的関数：レーンの中心に沿って自律走行する

2. 環境：AWS RoboMakerでホストされている3Dドライビングシミュレータ

3. 状態：右図のような、車のヘッドカメラで撮影した運転中のPOV画像

4. アクション：異なる角度で6つの別々のステアリングホイール位置（設定可能）

5. 報酬：[設定設定の例] 中心線に近づくことでプラスに、トラックから外れることにより大きなペナルティが課される。例えばステアリングペナルティを追加することなど、詳細の設定も可能



Amazon SageMaker Reinforcement Learning

End-to-end examples for classic RL and real-world RL application

Robotics

Industrial Control

HVAC

Autonomous Vehicles

Operators

Finance

Games

NLP

RL Environment to model real-world problems

AWS Simulation Environment

Amazon Sumerian

Amazon RoboMaker

Open Source Environment

EnergyPlus

RoboSchool

PyBullet

...

Custom Environments

Bring Your Own

Commercial Simulators

MATLAB& Simulink

Open AI Gym

RL Toolkits that provide RL agent algorithm implementations

RL-Coach

DQN

PPO

HER

Rainbow

...

RL-Ray RLLib

APEX

ES

IMPALA

A3C

...

Open AI Baselines

TRPO

GAIL

...

...

SageMaker Deep Learning Frameworks

TensorFlow

MXNet

PyTorch


Chainer

Training Options

Single Machine / Distributed

Local / Remote simulation

CPU/GPU Hardware

 SageMaker supported

 Customer BYO

Amazon SageMaker Reinforcement Learning

End-to-end examples for classic RL and real-world RL application

Robotics

Industrial Control

HVAC

Autonomous Vehicles

Operators

Finance

Games

NLP

RL Environment to model real-world problems

AWS Simulation Environment

Amazon Sumerian

Amazon RoboMaker

Open Source Environment

EnergyPlus

RoboSchool

PyBullet

...

Custom Environments

Bring Your Own

Commercial Simulators

MATLAB& Simulink

Open AI Gym

RL Toolkits that provide RL agent algorithm implementations

RL-Coach

DQN

PPO

HER

Rainbow

...

RL-Ray RLLib

APEX

ES

IMPALA

A3C

...

Open AI Baselines

TRPO

GAIL

...

...

SageMaker Deep Learning Frameworks

TensorFlow

MXNet

PyTorch


Chainer

Training Options

Single Machine / Distributed

Local / Remote simulation

CPU/GPU Hardware

 SageMaker supported

 Customer BYO

Amazon SageMaker Reinforcement Learning

End-to-end examples for classic RL and real-world RL application

Robotics

Industrial Control

HVAC

Autonomous Vehicles

Operators

Finance

Games

NLP

RL Environment to model real-world problems

AWS Simulation Environment

Amazon Sumerian

Amazon RoboMaker

Open Source Environment

EnergyPlus

RoboSchool

PyBullet

...

Custom Environments

Bring Your Own

Commercial Simulators

MATLAB& Simulink

Open AI Gym

RL Toolkits that provide RL agent algorithm implementations

RL-Coach

DQN

PPO

HER

Rainbow

...

RL-Ray RLLib

APEX

ES

IMPALA

A3C

...

Open AI Baselines

TRPO

GAIL

...

...

SageMaker Deep Learning Frameworks

TensorFlow

MXNet

PyTorch


Chainer


Training Options

Single Machine / Distributed

Local / Remote simulation

CPU/GPU Hardware

 SageMaker supported

 Customer BYO

Amazon SageMaker Reinforcement Learning

End-to-end examples for classic RL and real-world RL application

Robotics

Industrial Control

HVAC

Autonomous Vehicles

Operators

Finance

Games

NLP

RL Environment to model real-world problems

AWS Simulation Environment

Amazon Sumerian

Amazon RoboMaker

Open Source Environment

EnergyPlus

RoboSchool

PyBullet

...

Custom Environments

Bring Your Own

Commercial Simulators

MATLAB& Simulink

Open AI Gym

RL Toolkits that provide RL agent algorithm implementations

RL-Coach

DQN

PPO

HER

Rainbow

...

RL-Ray RLLib

APEX

ES

IMPALA

A3C

...

Open AI Baselines

TRPO

GAIL

...

...

SageMaker Deep Learning Frameworks

TensorFlow

MXNet

PyTorch


Chainer

Training Options

Single Machine / Distributed

Local / Remote simulation

CPU/GPU Hardware

 SageMaker supported

 Customer BYO

Amazon SageMaker Reinforcement Learning

End-to-end examples for classic RL and real-world RL application

Robotics

Industrial Control

HVAC

Autonomous Vehicles

Operators

Finance

Games

NLP

RL Environment to model real-world problems

AWS Simulation Environment

Amazon Sumerian

Amazon RoboMaker

Open Source Environment

EnergyPlus

RoboSchool

PyBullet

...

Custom Environments

Bring Your Own

Commercial Simulators

MATLAB& Simulink

Open AI Gym

RL Toolkits that provide RL agent algorithm implementations

RL-Coach

DQN

PPO

HER

Rainbow

...

RL-Ray RLLib

APEX

ES

IMPALA

A3C

...

Open AI Baselines

TRPO

GAIL

...

...

SageMaker Deep Learning Frameworks

TensorFlow

MXNet

PyTorch


Chainer

Training Options

Single Machine / Distributed

Local / Remote simulation

CPU/GPU Hardware

 SageMaker supported

 Customer BYO

Amazon SageMaker Reinforcement Learning

End-to-end examples for classic RL and real-world RL application

Robotics

Industrial Control

HVAC

Autonomous Vehicles

Operators

Finance

Games

NLP

RL Environment to model real-world problems

AWS Simulation Environment

Amazon Sumerian

Amazon RoboMaker

Open Source Environment

EnergyPlus

RoboSchool

PyBullet

...

Custom Environments

Bring Your Own

Commercial Simulators

MATLAB& Simulink

Open AI Gym

RL Toolkits that provide RL agent algorithm implementations

RL-Coach

DQN

PPO

HER

Rainbow

...

RL-Ray RLLib

APEX

ES

IMPALA

A3C

...

Open AI Baselines

TRPO

GAIL

...

...

SageMaker Deep Learning Frameworks

TensorFlow

MXNet

PyTorch


Chainer


Training Options

Single Machine / Distributed

Local / Remote simulation

CPU/GPU Hardware

 SageMaker supported

 Customer BYO

強化学習モデルの学習・デプロイ

```
from sagemaker.rl import RLEstimator, RLToolkit, RLFramework

# Train my estimator
rl_estimator = RLEstimator(entry_point='coach-train.py',
                           toolkit=RLToolkit.COACH,
                           toolkit_version='0.11.0',
                           framework=RLFramework.MXNET,
                           role='SageMakerRole',
                           train_instance_type='ml.c4.2xlarge',
                           train_instance_count=1)

rl_estimator.fit()

# Deploy my estimator to a SageMaker Endpoint and get a MXNetPredictor
predictor = rl_estimator.deploy(instance_type='ml.m4.xlarge',
                                initial_instance_count=1)

response = predictor.predict(data)
```

<https://github.com/aws/sagemaker-python-sdk/tree/master/src/sagemaker/rl>

学習用コンテナとシミュレーション用コンテナの分離

- 学習にはGPU，環境シミュレータにはCPU
- Estimatorを独立に作り，インスタンスを指定

```
primary_cluster_instance_type = "ml.p3.2xlarge"
primary_cluster_instance_count = 1

secondary_cluster_instance_type = "ml.c5.4xlarge"
secondary_cluster_instance_count = 2

total_cpus = 40 - 1 # Leave one for ray scheduler
total_gpus = 1
```

```
primary_cluster_estimator = RLEstimator(entry_point="train-%s.py" % roboschool_problem,
source_dir='src',
dependencies=["common/sagemaker_rl"],
image_name=gpu_image_name,
role=role,
train_instance_type=primary_cluster_instance_type,
train_instance_count=primary_cluster_instance_count,
output_path=s3_output_path,
base_job_name=job_name_prefix,
metric_definitions=metric_definitions,
train_max_run=int(3600 * .5), # Maximum runtime in seconds
hyperparameters={
    "s3_prefix": s3_prefix, # Important for syncing
    "s3_bucket": s3_bucket, # Important for syncing
    "aws_region": boto3.Session().region_name, # Important for S3 connection
    "rl_cluster_type": "primary", # Important for syncing
    "rl_num_instances_secondary": secondary_cluster_instance_count, # Important for syncing
    "rl.training.config.num_workers": total_cpus,
    "rl.training.config.train_batch_size": 20000,
    "rl.training.config.num_gpus": total_gpus,
},
subnets=default_subnets, # Required for VPC mode
security_group_ids=default_security_groups # Required for VPC mode
)

primary_cluster_estimator.fit(wait=False)
primary_job_name = primary_cluster_estimator.latest_training_job.job_name
print("Primary Training job: %s" % primary_job_name)
```

Primary cluster with GPU

https://github.com/aws-labs/amazon-sagemaker-examples/blob/master/reinforcement_learning/rl_roboschool_ray/rl_roboschool_ray_distributed.ipynb

```
secondary_cluster_estimator = RLEstimator(entry_point="train-%s.py" % roboschool_problem,
source_dir='src',
dependencies=["common/sagemaker_rl"],
image_name=cpu_image_name,
role=role,
train_instance_type=secondary_cluster_instance_type,
train_instance_count=secondary_cluster_instance_count,
output_path=s3_output_path,
base_job_name=job_name_prefix,
metric_definitions=metric_definitions,
train_max_run=3600, # Maximum runtime in seconds
hyperparameters={
    "s3_prefix": s3_prefix, # Important for syncing
    "s3_bucket": s3_bucket, # Important for syncing
    "aws_region": boto3.Session().region_name, # Important for S3 connection
    "rl_cluster_type": "secondary", # Important for syncing
},
subnets=default_subnets, # Required for VPC mode
security_group_ids=default_security_groups # Required for VPC mode
)

secondary_cluster_estimator.fit(wait=False)
secondary_job_name = secondary_cluster_estimator.latest_training_job.job_name
print("Secondary Training job: %s" % secondary_job_name)
```

Secondary cluster with CPU



SageMaker Reinforcement Learning 対応フレームワーク

Dependencies	Coach 0.10.1	Coach 0.11.0	Ray 0.5.3
Python	3.6	3.5 (MXNet) 3.6 (TensorFlow)	3.6
CUDA (GPU image only)	9.0	9.0	9.0
DL Framework	TensorFlow-1.11.0	MXNet-1.3.0 TensorFlow-1.11.0	TensorFlow-1.11.0
gym	0.10.5	0.10.5	0.10.5

<https://github.com/aws/sagemaker-python-sdk/tree/master/src/sagemaker/rl#distributed-rl-training>

※ 2019年2月13日時点の情報です


AWS DeepRacer

Amazon.com にて\$399.00, 2019/3/6 発売開始予定

仕様

- 1/18 スケールラジコンカー
- Intel Atom プロセッサ
- 4Mピクセル, 1080p カメラ
- WiFi (802.11ac)
- Compute用バッテリー (>2h)と、
モーター用バッテリー(>15m)
- Ubuntu 16.04LTS, ROS (Robot Operat
System), OpenVino

<https://aws.amazon.com/deepracer/>



Click image to open expanded view

AWS DeepRacer – Fully autonomous 1/18th scale race car for developers
by [Amazon Web Services](#)
[13 answered questions](#)

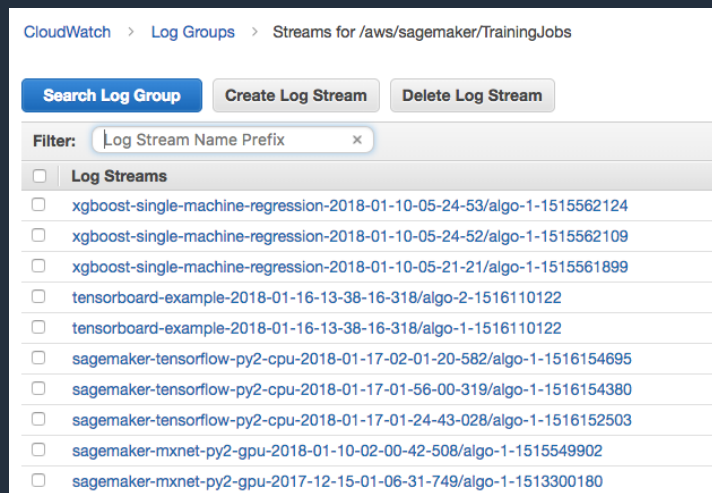
Price: **\$399.00**

This item will be released on March 6, 2019.
Pre-order now.
This item does not ship to **Japan**. Please check other sellers who may ship internationally. [Learn more](#)
Ships from and sold by Amazon Digital Services LLC.

- AWS DeepRacer is the fastest way for developers to get rolling with machine learning, literally. Get hands-on with a fully autonomous 1/18th scale race car driven by reinforcement learning, 3D racing simulator, and a global racing league.
- Car - AWS DeepRacer is an autonomous 1/18th scale race car designed to test reinforcement learning (RL) models by racing on a physical track.
- Simulator - Build models in Amazon SageMaker and train, test, and iterate quickly and easily on the track in the AWS DeepRacer console and 3D racing simulator.
- League - Compete in the world's first global, autonomous racing league, to race for prizes and glory and a chance to advance to the AWS DeepRacer Grand Final to win the coveted AWS DeepRacer Cup.
- Build RL Models: You can build your own RL model for AWS DeepRacer using the AWS DeepRacer 3D racing simulator. Building a model requires basic Python programming skills.

学習ジョブの評価

- 学習ジョブの実行時のログは、すべて CloudWatch Logs に出力される
- 自作コンテナを使う場合は、標準出力に出したものがそのまま CloudWatch Logs に送られる
- モデル評価等は、すべてログに出力して、あとから集計

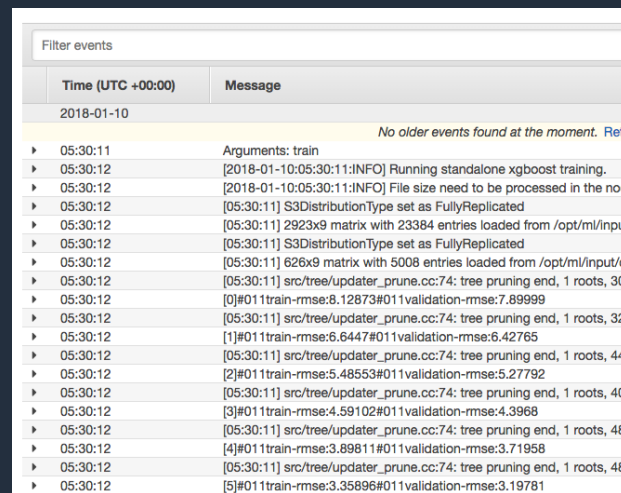


CloudWatch > Log Groups > Streams for /aws/sagemaker/TrainingJobs

Search Log Group Create Log Stream Delete Log Stream

Filter: Log Stream Name Prefix x

Log Streams
<input type="checkbox"/> xgboost-single-machine-regression-2018-01-10-05-24-53/algo-1-1515562124
<input type="checkbox"/> xgboost-single-machine-regression-2018-01-10-05-24-52/algo-1-1515562109
<input type="checkbox"/> xgboost-single-machine-regression-2018-01-10-05-21-21/algo-1-1515561899
<input type="checkbox"/> tensorboard-example-2018-01-16-13-38-16-318/algo-2-1516110122
<input type="checkbox"/> tensorboard-example-2018-01-16-13-38-16-318/algo-1-1516110122
<input type="checkbox"/> sagemaker-tensorflow-py2-cpu-2018-01-17-02-01-20-582/algo-1-1516154695
<input type="checkbox"/> sagemaker-tensorflow-py2-cpu-2018-01-17-01-56-00-319/algo-1-1516154380
<input type="checkbox"/> sagemaker-tensorflow-py2-cpu-2018-01-17-01-24-43-028/algo-1-1516152503
<input type="checkbox"/> sagemaker-mxnet-py2-gpu-2018-01-10-02-00-42-508/algo-1-1515549902
<input type="checkbox"/> sagemaker-mxnet-py2-gpu-2017-12-15-01-06-31-749/algo-1-1513300180



Filter events

Time (UTC +00:00)	Message
2018-01-10	No older events found at the moment. Retr...
05:30:11	Arguments: train
05:30:12	[2018-01-10:05:30:11:INFO] Running standalone xgboost training.
05:30:12	[2018-01-10:05:30:11:INFO] File size need to be processed in the nod...
05:30:12	[05:30:11] S3DistributionType set as FullyReplicated
05:30:12	[05:30:11] 2923x9 matrix with 23384 entries loaded from /opt/ml/input...
05:30:12	[05:30:11] S3DistributionType set as FullyReplicated
05:30:12	[05:30:11] 626x9 matrix with 5008 entries loaded from /opt/ml/input/d...
05:30:12	[05:30:11] src/tree/updater_prune.cc:74: tree pruning end, 1 roots, 30
05:30:12	[0]#011train-rmse:8.12873#011validation-rmse:7.89999
05:30:12	[05:30:11] src/tree/updater_prune.cc:74: tree pruning end, 1 roots, 32
05:30:12	[1]#011train-rmse:6.6447#011validation-rmse:6.42765
05:30:12	[05:30:11] src/tree/updater_prune.cc:74: tree pruning end, 1 roots, 44
05:30:12	[2]#011train-rmse:5.48553#011validation-rmse:5.27792
05:30:12	[05:30:11] src/tree/updater_prune.cc:74: tree pruning end, 1 roots, 40
05:30:12	[3]#011train-rmse:4.59102#011validation-rmse:4.3968
05:30:12	[05:30:11] src/tree/updater_prune.cc:74: tree pruning end, 1 roots, 48
05:30:12	[4]#011train-rmse:3.89811#011validation-rmse:3.71958
05:30:12	[05:30:11] src/tree/updater_prune.cc:74: tree pruning end, 1 roots, 48
05:30:12	[5]#011train-rmse:3.35896#011validation-rmse:3.19781

メトリクスのモニタリング

- 学習時のモデルの精度をグラフ化できる
- 学習が上手く行っているかどうかを確認し、必要に応じて学習を止める判断に利用できる
- CloudWatch のメトリクスダッシュボードで、グラフの可視化



学習スクリプトのメトリクス定義

- 自前のスクリプトについては、メトリクス正規表現で定義

```
from sagemaker.tuner import HyperparameterTuner, ContinuousParameter

# Configure HyperparameterTuner
my_tuner = HyperparameterTuner(estimator=my_estimator, # previously-configured Estimator object
                               objective_metric_name='validation-accuracy',
                               hyperparameter_ranges={'learning-rate': ContinuousParameter(0.05, 0.06)},
                               metric_definitions=[{'Name': 'validation-accuracy', 'Regex':
                                                    'validation-accuracy=(\d\.\d+)'}],
                               max_jobs=100,
                               max_parallel_jobs=10)

# Start hyperparameter tuning job
my_tuner.fit({'train': 's3://my_bucket/my_training_data', 'test': 's3://my_bucket/my_testing_data'})

# Deploy best model
my_predictor = my_tuner.deploy(initial_instance_count=1, instance_type='ml.m4.xlarge')

# Make a prediction against the SageMaker endpoint
response = my_predictor.predict(my_prediction_data)
```

Q&A

ご質問については、AWS Japan Blog
「<https://aws.amazon.com/jp/blogs/news/>」
にて後日掲載します

公式Twitter/Facebook AWSの最新情報をお届けします



@awscloud_jp



検索

もしくは
<http://on.fb.me/1vR8yWm>

最新技術情報、イベント情報、お役立ち情報、
お得なキャンペーン情報などを日々更新しています！

AWS Well-Architected 個別技術相談会

毎週”W-A個別技術相談会”を実施中

- AWSのソリューションアーキテクト(SA)に
対策などを相談することも可能

• 申込みはイベント告知サイトから

(<https://aws.amazon.com/jp/about-aws/events/>)

AWS イベント で[検索]

