aws 認定デバイス IoT

Amazon SageMaker JumpStartの機械学習モデルをAE2100上へ実装・推論

沖電気工業株式会社

140th Anniversary Towards 2031

アジェンダ

1. デモ環境および事前準備
2. SageMaker JumpStartのファインチューニングで分類モデルを作成
3. OpenVINOのModel OptimizerでIRへ変換
4. 推論用デモプログラムの準備
5. AE2100での推論実行
6. まとめ

1. デモ環境および事前準備

2. SageMaker JumpStartのファインチューニングで分類モデルを作成

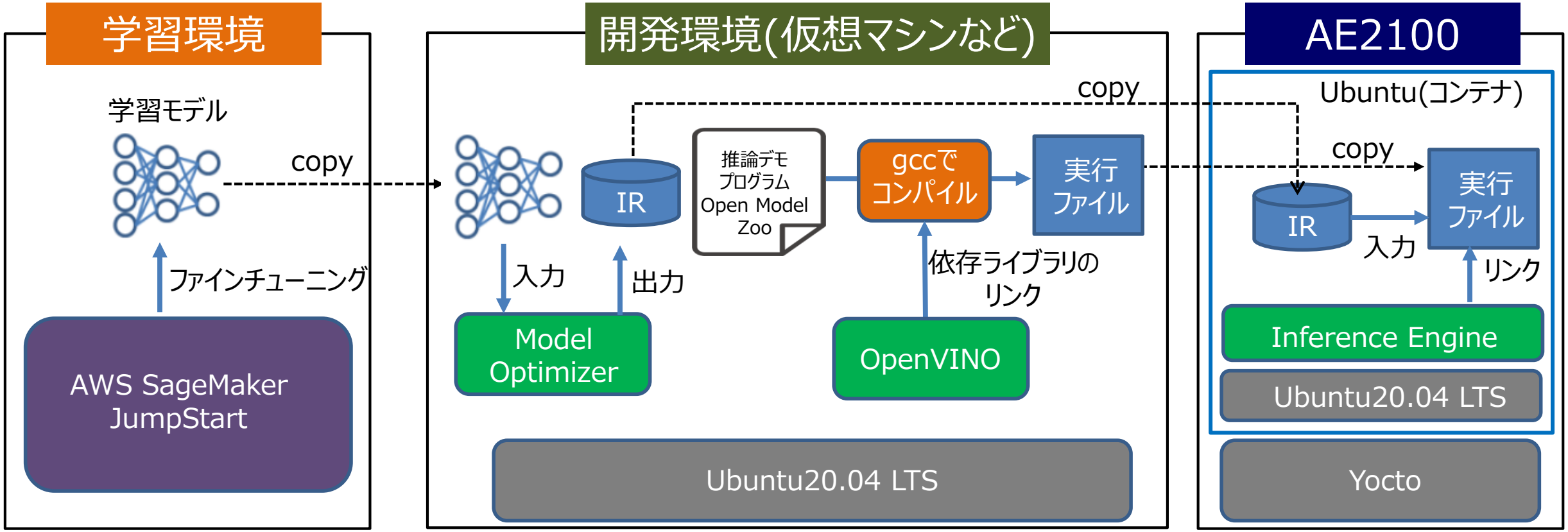
3. OpenVINOのModel OptimizerでIRへ変換

4. 推論用デモプログラムの準備

5. AE2100での推論実行

6. まとめ

今回の体験での各種環境

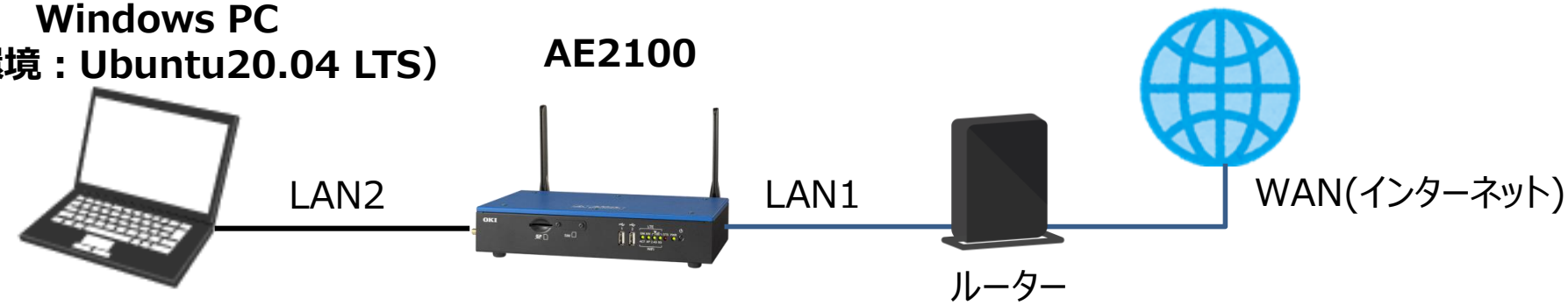


開発環境 OS：Ubuntu20.04 LTS　　OpenVINO：2021.4.1 LTS

AE2100 本体ファーム：V3.6.0（HDDL Daemon：OpenVINO 2021.4.1 LTS）
標準コンテナ：OpenVINO有（Ubuntu20.04版）2021年10月版

機器の接続構成と体験前にご準備いただく内容

【補足】
開発環境はVirtualBoxやVmwareでWindows上に構築できます。
また、AWSのEC2上に構築することも可能です。



【開発環境】
- 開発環境の構築

【AE2100】
- HDDL Daemonの有効化
- コンテナのセットアップ・起動
- 依存パッケージのダウンロード

【Windows PC】
- Xlaunchのインストール

【SDK取扱説明書（DeepLearnning編）】を参照

【Qiitaの記事】を参照

OKI AI エッジコンピューター「AE2100」でOpenVINOのデモプログラムを動かしてみよう Ubuntuコンテナ版 (1)、(2)
https://qiita.com/TWAT/items/7398105a9e64178a84d7
https://qiita.com/TWAT/items/e7cd34f8c97f895c39b2

1. デモ環境および事前準備

2. SageMaker JumpStartのファインチューニングで分類モデルを作成

3. OpenVINOのModel OptimizerでIRへ変換

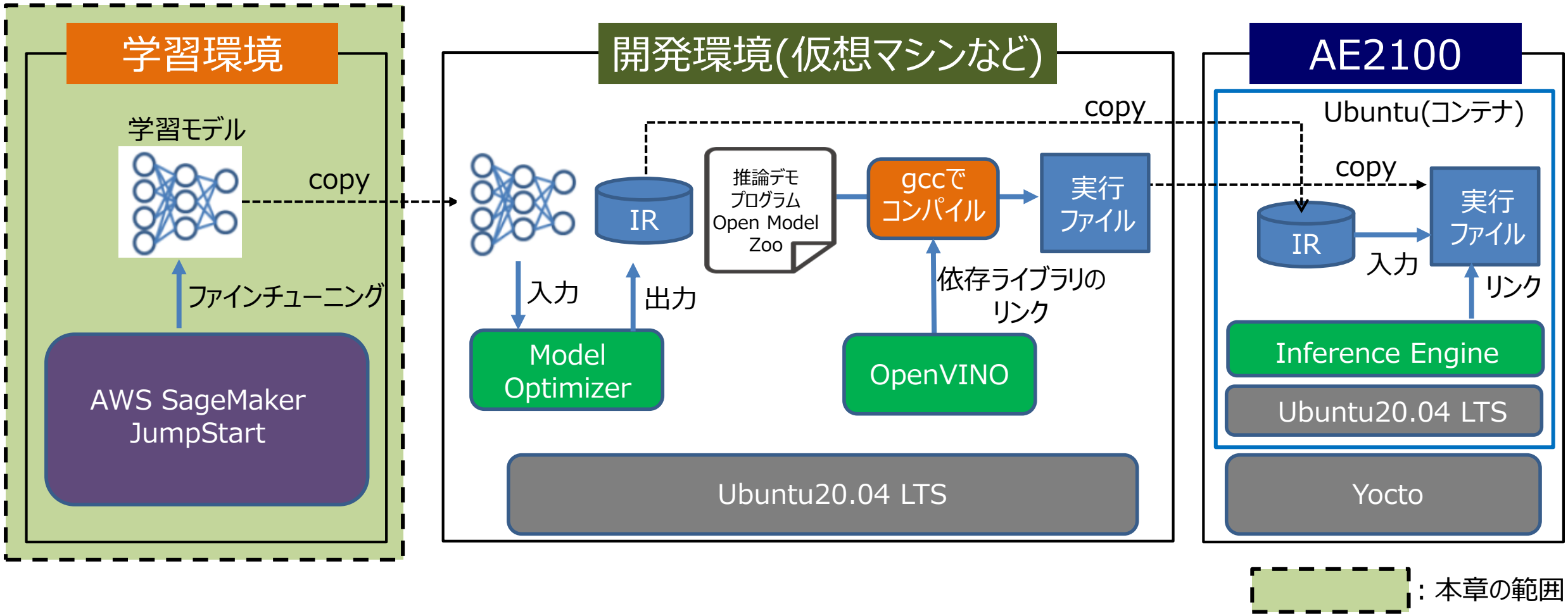
4. 推論用デモプログラムの準備

5. AE2100での推論実行

6. まとめ

SageMaker JumpStartのファインチューニングで分類モデルを作成

AWS SageMaker JumpStartでTensorFlowの画像分類モデルをファインチューニング
→「model.tar.gz」



体験で使用するモデルと学習・推論用のデータの用意

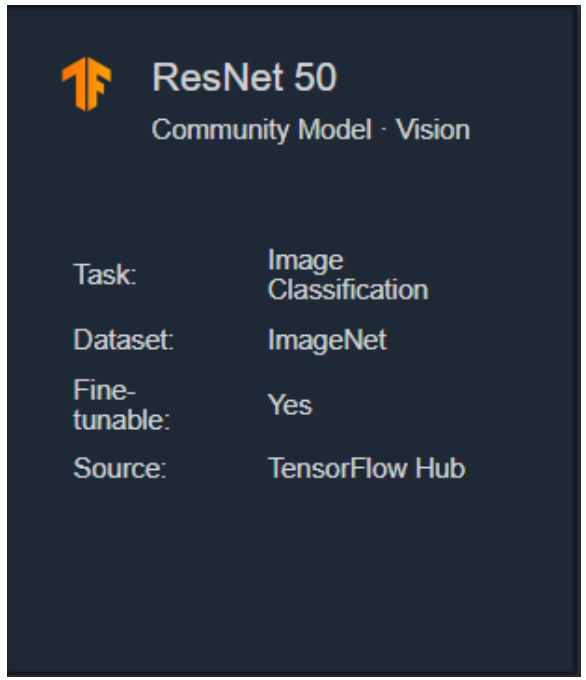
学習環境

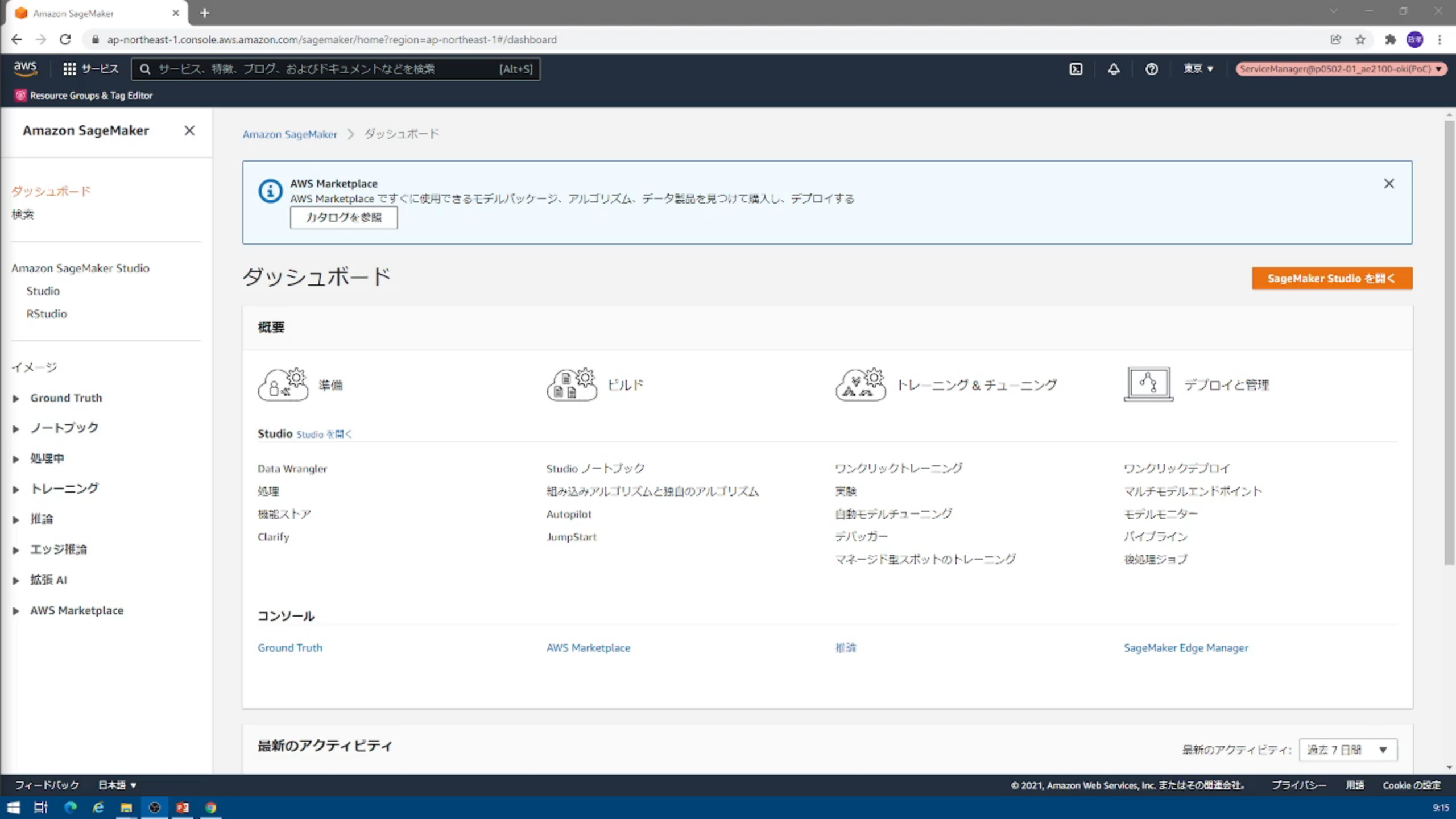
- ファインチューニングに使用する画像分類モデル
  ResNet 50（TensorFlow）

- 今回作成する分類モデル
  ソース、ケチャップ、マヨネーズの分類モデル

- 学習・推論用データの用意
  ソース、マヨネーズ、ケチャップをiPhoneで各30枚撮影
  →学習用：各20枚、推論用：各10枚

- ハイパーパラメータ
  Epochs：100、Learnning Rate：0.01、Batch Size：4




Amazon SageMaker
ap-northeast-1.console.aws.amazon.com/sagemaker/home?region=ap-northeast-1#/dashboard
サービス
サービス、特徴、ブログ、およびドキュメントなどを検索
[Alt+S]
東京
Resource Groups & Tag Editor
Amazon SageMaker
ダッシュボード
検索
Amazon SageMaker Studio
Studio
RStudio
イメージ
Ground Truth
ノートブック
処理中
トレーニング
推論
エッジ推論
拡張 AI
AWS Marketplace
Amazon SageMaker > ダッシュボード
AWS Marketplace
AWS Marketplace ですぐに使用できるモデルパッケージ、アルゴリズム、データ製品を見つけて購入し、デプロイする
カタログを参照
ダッシュボード
SageMaker Studio を開く
概要
準備
ビルド
トレーニング & チューニング
デプロイと管理
Studio Studio を開く
Data Wrangler
処理
機能ストア
Clarify
Studio ノートブック
組み込みアルゴリズムと独自のアルゴリズム
Autopilot
JumpStart
ワンクリックトレーニング
実験
自動モデルチューニング
デバッガー
マネージド型スポットのトレーニング
ワンクリックデプロイ
マルチモデルエンドポイント
モデルモニター
パイプライン
後処理ジョブ
コンソール
Ground Truth
AWS Marketplace
推論
SageMaker Edge Manager
最新のアクティビティ
最新のアクティビティ:
過去 7 日間
フィードバック
日本語
© 2021, Amazon Web Services, Inc. またはその関連会社。
プライバシー
用語
Cookie の設定
9:15

1. デモ環境および事前準備

2. SageMaker JumpStartのファインチューニングで分類モデルを作成

3. OpenVINOのModel OptimizerでIRへ変換

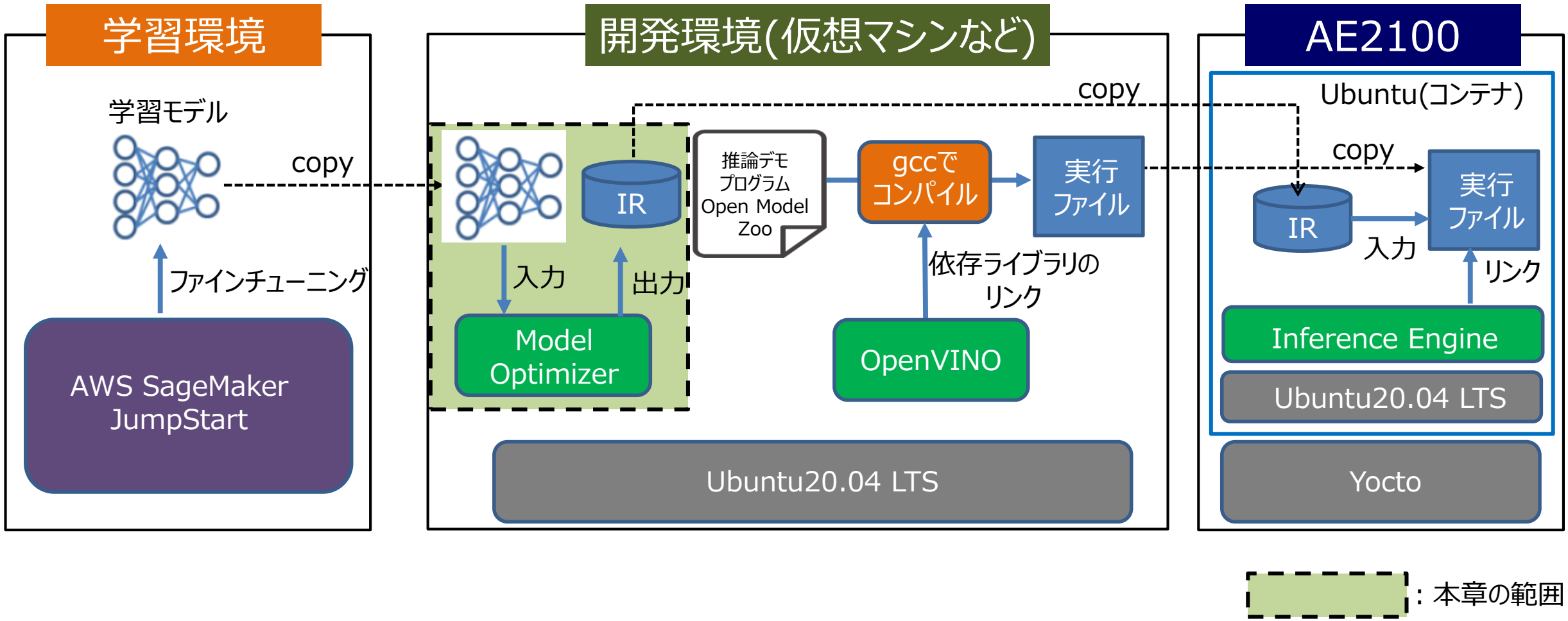
4. 推論用デモプログラムの準備

5. AE2100での推論実行

6. まとめ

OpenVINOのModel OptimizerでIRへ変換

OpenVINOツールキット付属の**Model Optimizerにより、IR形式へ変換**
→「saved_model.pb」、「saved_model.xml」



Model Optimizerでの変換手順

AWS S3からダウンロードしたモデル**「model.tar.gz」を開発環境で展開**します。

```
# tar xzf model.tar.gz
```

→「1」、「code」というディレクトリと「class_label_to_prediction_index.json」というファイルが生成

Model OptimizerはPythonにより実行するため、**Pythonの仮想環境を起動**します。

```
# source /opt/intel/openvino_2021/deployment_tools/model_optimizer/venv/bin/activate
```

下記のコマンドで**mo_tf.pyを実行し、モデルファイルをIRへ変換**します。

```
# python3 /opt/intel/openvino_2021/deployment_tools/model_optimizer/mo_tf.py ¥
> --data_type FP16 ¥
> --reverse_input_channels ¥
> --saved_model_dir 1 ¥
> --input_shape=[1,224,224,3] ¥
> --scale_values=[255.0]
```

→「saved_model.xml」、「saved_model.bin」、「saved_model.mapping」というファイルが生成

Model Optimizerでの変換パラメータ

Model Optimizerで変換する際のパラメータは使用するモデルや推論用のデモプログラムに何を使用するかによって異なるため、変換パラメータを事前に調べる必要があります。

引数	説明	今回のデモ
--data_type {FP16,FP32,half,float}	IRのデータ型	FP16 →**Myriad X VPUで推論を行う**ため、データ型としてFP16を指定
--reverse_input_channels	入力チャネルの並びをRGBからBGRへ(またはその逆へ)入れ替える	**inference.py(※)にRGBである旨**の記述有 **classification_demoはBGRチャネルの順序**での入力
--saved_model_dir	Saved Modelが格納されているディレクトリ名	1 →**saved_model.pbの格納ディレクトリ名**を指定
--input_shape INPUT_SHAPE	入力データの形状	[1,224,224,3] →**inference.py**に記述有
--scale_values SCALE_VALUES	入力画像のチャネルごとのスケール値	[255.0] →**inference.py**に記述有

※ model.tar.gz内に含まれる「code」フォルダ内に格納

【参考】
https://docs.openvino.ai/2021.4/openvino_docs_MO_DG_prepare_model_convert_model_Converting_Model.html
https://docs.openvino.ai/2021.4/openvino_docs_MO_DG_prepare_model_convert_model_Convert_Model_From_TensorFlow.html
https://docs.openvino.ai/2021.4/omz_demos_classification_demo_cpp.html

詳細は講演動画を参照

model.tar.gzを展開

```
# tar xzf model.tar.gz
```

Pythonの仮想環境を起動し、ModelOptimizerを実行

```
# source  /opt/intel/openvino_2021/deployment_tools/model_optimizer/venv/bin/activate
# python3 /opt/intel/openvino_2021/deployment_tools/model_optimizer/mo_tf.py ¥
> --data_type FP16 ¥
> --reverse_input_channels ¥
> --saved_model_dir 1 ¥
> --input_shape=[1,224,224,3] ¥
> --scale_values=[255.0]
```

1. デモ環境および事前準備

2. SageMaker JumpStartのファインチューニングで分類モデルを作成

3. OpenVINOのModel OptimizerでIRへ変換

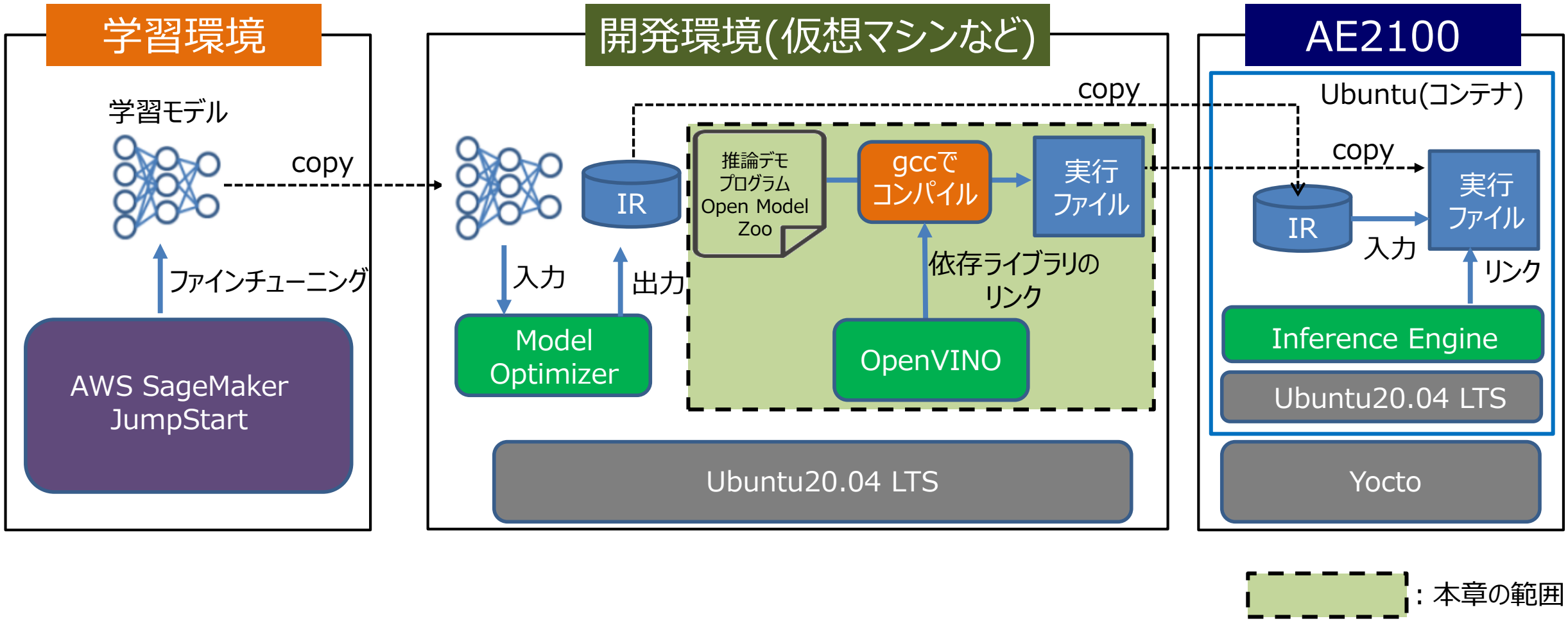
4. 推論用デモプログラムの準備

5. AE2100での推論実行

6. まとめ

推論用デモプログラムの準備

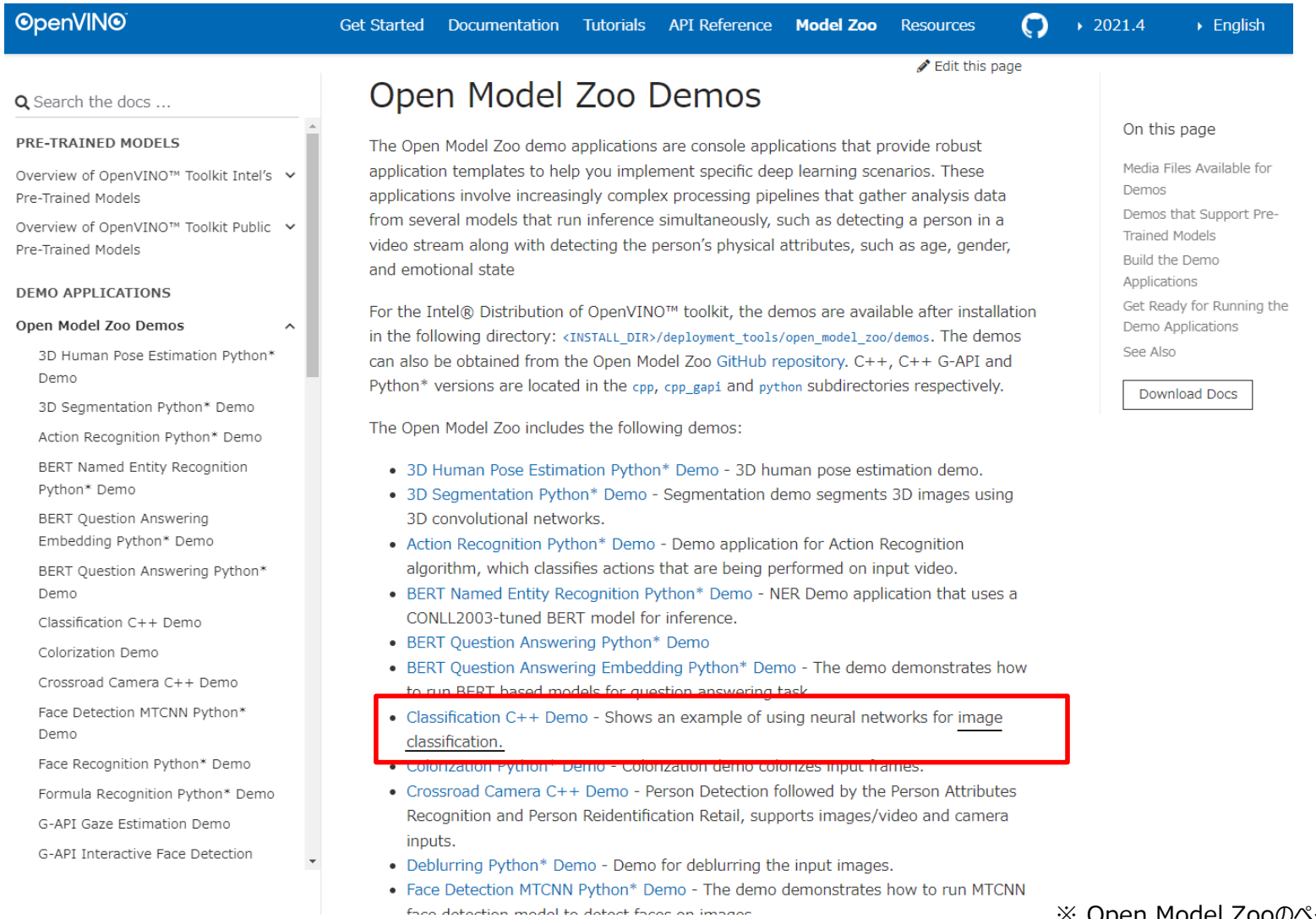
Open Model Zooのデモプログラムをビルドし、**ラベルファイルと推論用画像**を用意します。
→「classification_demo」、「imagenet_2012.txt」、推論用画像



デモプログラムの選定

開発環境

Open Model Zoo：最適化済みでオープンソースの学習済みモデルやデモプログラム等を用意



https://docs.openvino.ai/2021.4/omz_demos.html

今回は「Classification C++ Demo」を使用

→個別ページにて以下の情報が記載

- アプリケーションの概要
- 対応モデル
- アプリケーションの実行方法
- ラベルファイルのフォーマット etc

※ Open Model Zooのページ抜粋

「Classification C++ Demo」の用意

OpenVINOの環境変数を設定します。

```
# source /opt/intel/openvino/bin/setupvars.sh
```

Open Model Zooのデモプログラムがあるディレクトリに移動し、**Demosをビルド**します。

```
# cd /opt/intel/openvino/inference_engine/demos
# ./build_demos.sh
```

→「omz_demos_build」というディレクトリが生成

omz_demos_build/intel64/Releaseに「**classification_demo**」が格納されます。

```
# cd
# cd omz_demos_build/intel64/Release/
# ls
classification_demo                multi_channel_face_detection_demo    crossroad_camera_demo
multi_channel_human_pose_estimation_demo    gaze_estimation_demo
multi_channel_object_detection_demo_yolov3    gaze_estimation_demo_gapi    object_detection_demo
human_pose_estimation_demo    pedestrian_tracker_demo    image_processing_demo
security_barrier_camera_demo    interactive_face_detection_demo    segmentation_demo
interactive_face_detection_demo_gapi    smart_classroom_demo    lib
social_distance_demo    mask_rcnn_demo    text_detection_demo
```

ラベルファイルの用意

Search the docs ...

PRE-TRAINED MODELS

Overview of OpenVINO™ Toolkit Intel's Pre-Trained Models

Overview of OpenVINO™ Toolkit Public Pre-Trained Models

DEMO APPLICATIONS

Open Model Zoo Demos

- 3D Human Pose Estimation Python* Demo
- 3D Segmentation Python* Demo
- Action Recognition Python* Demo
- BERT Named Entity Recognition Python* Demo
- BERT Question Answering Embedding Python* Demo
- BERT Question Answering Python* Demo
- **Classification C++ Demo**
- Colorization Demo
- Crossroad Camera C++ Demo
- Face Detection MTCNN Python* Demo
- Face Recognition Python* Demo
- Formula Recognition Python* Demo

Required Files

If you want to see classification results, you must use "-gt" and "-labels" flags to specify two .txt files containing lists of classes and labels.

"The ground truth" file is used for matching image file names with correct object classes.

It has the following format:

```
./ILSVRC2012_val_00000001.JPEG 65
./ILSVRC2012_val_00000002.JPEG 970
./ILSVRC2012_val_00000003.JPEG 230
...
```

Class index values must be in range from 0 to 1000. If you want to use "other" class, which is supported only by a small subset of models, specify it with -1 index.

"Labels" file contains the list of human-readable labels, one line for each class.

Please note that you should use `<omz_dir>/data/dataset_classes/imagenet_2015.txt` labels file with the following models:

- googlenet-v2
- se-inception
- se-resnet-101
- se-resnet-152
- se-resnet-50
- se-resnext-101
- se-resnext-50

and `<omz_dir>/data/dataset_classes/imagenet_2012.txt` labels file with all other models supported by the demo.

On this page

- How It Works
- Preparing to Run
  - Supported Models
  - Required Files
- Running
- Demo Output
- See Also

Download Docs

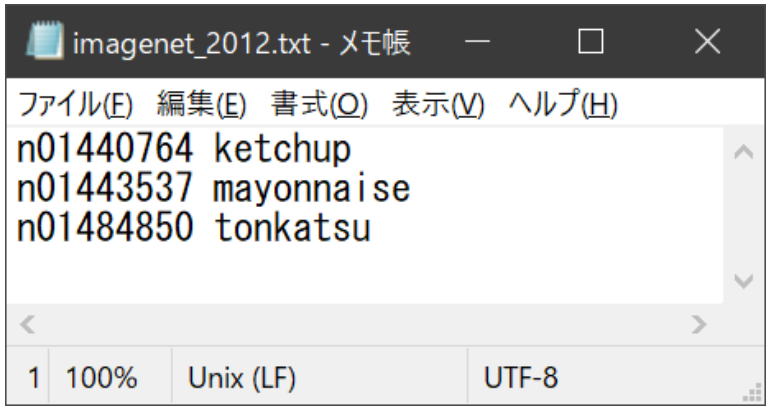
ラベルファイルは
「imagenet_2012.txt」を使用すること

※ Open Model Zooのページ抜粋

「imagenet_2012.txt」をローカルにコピーし、**ラベルファイルの中身を書き換え**

※ ラベルの記載順は、以下に従うこと
「class_label_to_prediction_index.json」

「code」ディレクトリに含まれるファイル



imagenet_2012.txtをコピーし、ラベルファイルの中身を書き換えます。

```
# cd /opt/intel/openvino/deployment_tools/open_model_zoo/data/dataset_classes/
# cp imagenet_2012.txt /home/shika/
```

tarアーカイブの作成

前項までで用意した以下のファイルと推論用の画像データ（ディレクトリ）を同じディレクトリに格納してください。
格納したディレクトリのtarアーカイブを作成します。

- saved_model.bin
- saved_model.xml
- classification_demo
- imagenet_2012.txt
- 推論用の画像ディレクトリ
  →「inference_sample」というディレクトリ」に「tonkatsu」、「ketchup」、「mayonnaise」の
  ディレクトリを作成し、それぞれ10枚のデータを格納しています。

```
# tar cvf sources_classification.tar sources_classification
```

詳細は講演動画を参照

OpeVINOの環境変数を設定

```
# source /opt/intel/openvino/bin/setupvars.sh
```

Demosをビルド

```
# cd /opt/intel/openvino/inference_engine/demos
# ./build_demos.sh
```

「classification_demo」が格納されていることを確認

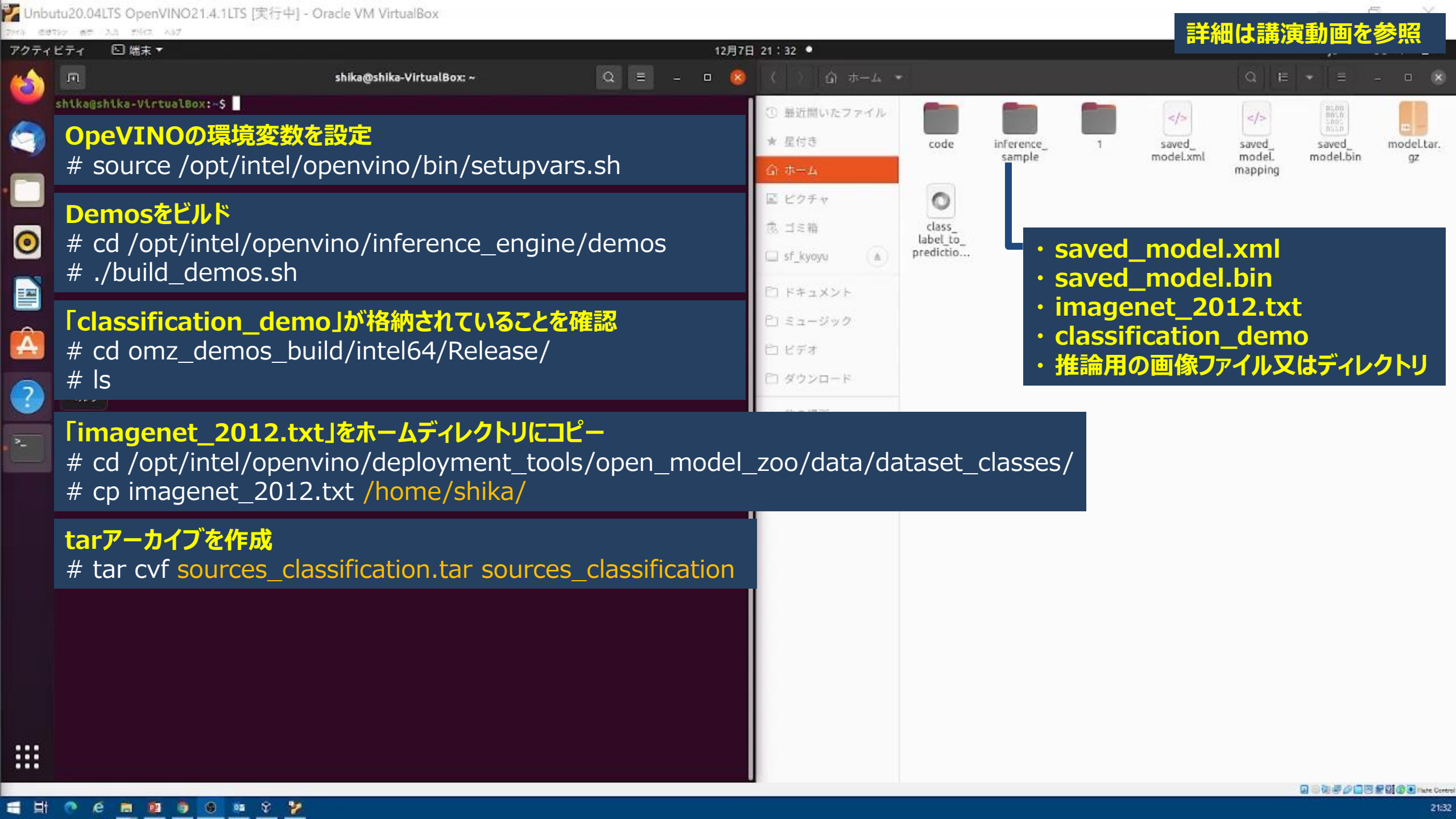
```
# cd omz_demos_build/intel64/Release/
# ls
```

「imagenet_2012.txt」をホームディレクトリにコピー

```
# cd /opt/intel/openvino/deployment_tools/open_model_zoo/data/dataset_classes/
# cp imagenet_2012.txt /home/shika/
```

tarアーカイブを作成

```
# tar cvf sources_classification.tar sources_classification
```



- **saved_model.xml**
- **saved_model.bin**
- **imagenet_2012.txt**
- **classification_demo**
- **推論用の画像ファイル又はディレクトリ**

1. デモ環境および事前準備

2. SageMaker JumpStartのファインチューニングで分類モデルを作成

3. OpenVINOのModel OptimizerでIRへ変換

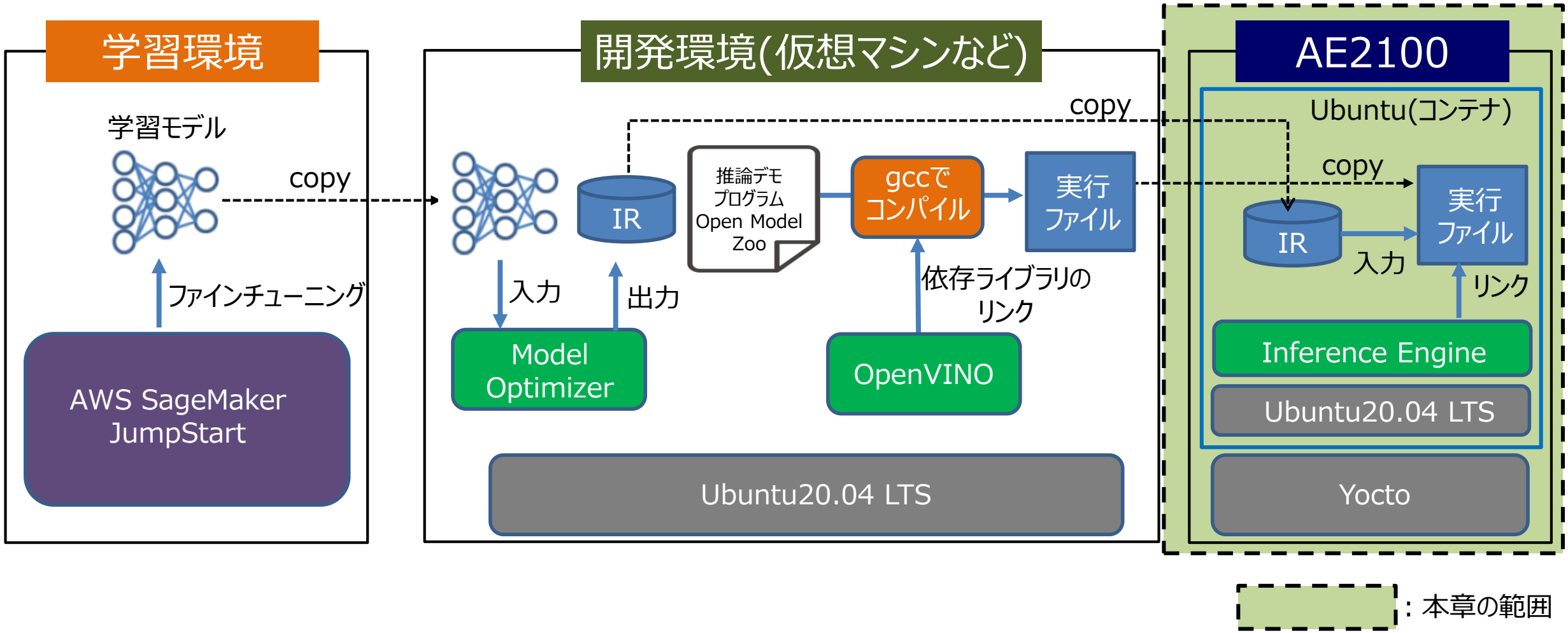
4. 推論用デモプログラムの準備

5. AE2100での推論実行

6. まとめ

AE2100での推論実行

3章、4章で作成したtarアーカイブをAE2100にコピーし、**コンテナ内でclassification_demoを実行**



AE2100での事前準備

HDDL Daemonが有効になっていることを確認してください。

```
# systemctl status hddldaemon.service
```

→最後尾に「systemd[1]: Started Intel OpenVINO HDDL Daemon.」が表示されていれば HDDL デバイスが有効化され、VPU での推論が可能な状態になっています。

コンテナを起動しておいてください。今回のデモではSDKマニュアル(DeepLearnning編)に従って用意した標準コンテナ「ubuntu-openvino」を使用しています。

```
# docker start ubuntu-openvino
```

デモ実行にffmpegやGTK+が必要なため、コンテナに**依存パッケージをインストール**しておいてください。
※インターネットへの接続が必要です。

```
root@ae2100:~# docker exec -it ubuntu-openvino /bin/bash
# cd
# cd /opt/intel/openvino/install_dependencies
# apt-get clean
# ./install_openvino_dependencies.sh
```

AE2100での推論実行(1)

SCP等でAE2100のホストOSに4章で作成した**tarアーカイブを転送し、コンテナにコピー**します。

```
root@ae2100:~# docker cp sources_classification.tar ubuntu-openvino:/root/
```

以下のコマンドでコンテナに入ります。

```
root@ae2100:~# docker exec -it ubuntu-openvino /bin/bash
```

コンテナ内でtarファイルを展開します。

```
# cd
# tar xvf sources_classification.tar
```

Windows PCで「Xlaunch」を起動し、画面エクスポート設定を行います。

```
# export DISPLAY=192.168.100.2:0.0
```

※Xlaunxhのインストール方法や起動方法は以下のURLのQiitaの記事をご参照ください。
【OKI AI エッジコンピューター「AE2100」でOpenVINOのサンプルプログラムを動かしてみよう Ubuntuコンテナ版 (1)】
https://qiita.com/TWAT/items/7398105a9e64178a84d7

AE2100での推論実行(2)

以下のコマンドでデモプログラムを実行します。

```
# cd  sources_classification
# ./classification_demo -m saved_model.xml -i inference_sample/tonkatsu/ -d HDDL -labels imagenet_2012.txt -nt 3
```

オプション	説明	今回のデモ
-m	xmlファイルへのパスを指定	saved_model.xml
-i	推論用の画像ファイル（ディレクトリ）へのパスを指定	推論用のトンカツソースの画像ディレクトリへのパス （マヨネーズ、ケチャップを推論する場合も同様）
-d	推論に使用するデバイスを指定	Myriad X VPU：HDDL CPU：CPU GPU：GPU
-labels	ラベルファイルへのパスを指定	imagenet_2012.txt
-nt	結果の出力数	3種類の分類モデルなので「3」を指定 【注意】デフォルトは「5」になっており、4種類以下の分類モデルの場合はエラー発生

※アプリケーション実行のオプションは以下のURLに掲載されています。
https://docs.openvino.ai/2021.4/omz_demos_classification_demo_cpp.html

詳細は講演動画を参照

tarアーカイブをコンテナにコピー
```
# docker cp sources_classification.tar ubuntu-openvino:/root/
```

コンテナに移動
```
# docker exec -it ubuntu-openvino /bin/bash
```

tarアーカイブを解凍
```
# cd
# tar xvf sources_classification.tar
```

「Xlaunch」を起動し、画面エクスポート設定を行います
```
# export DISPLAY=192.168.100.2:0.0
```

デモプログラムを実行（トンカツソース）
```
# cd  sources_classification
# ./classification_demo -m saved_model.xml -i inference_sample/tonkatsu/ -d HDDL -labels imagenet_2012.txt -nt 3
```

デモプログラムを実行（ケチャップ）
```
# ./classification_demo -m saved_model.xml -i inference_sample/ketchup/ -d HDDL -labels imagenet_2012.txt -nt 3
```

デモプログラムを実行（マヨネーズ）
```
# ./classification_demo -m saved_model.xml -i inference_sample/mayonnaise/ -d HDDL -labels imagenet_2012.txt -nt 3
```

1. デモ環境および事前準備
2. SageMaker JumpStartのファインチューニングで分類モデルを作成
3. OpenVINOのModel OptimizerでIRへ変換
4. 推論用デモプログラムの準備
5. AE2100での推論実行

6. まとめ

まとめ

- **AWS SageMaker JumpStartで既存モデルのファインチューニングが可能**
  →独自のAIモデルを高価なサーバーを用意しなくても簡単に作れる！

- **OpenVINO上で動作させるためには、Model Optimizerでの変換が必要**
  →慣れないと難しいかもしれないが、OKIではIntel共催セミナーやQiita等の開発用のコンテンツを多数用意！

- **Open Model Zooを使いこなせば簡単にAIモデルの推論まで実行可能**
  →画像分類モデルに限らず、顔認識や文字認識も簡単に可能！

- **AE2100はAWS IoT Greengrassに対応**
  →SageMaker JumpStart以外にもAWSの色々なサービスと連携可能！

Open up your dreams