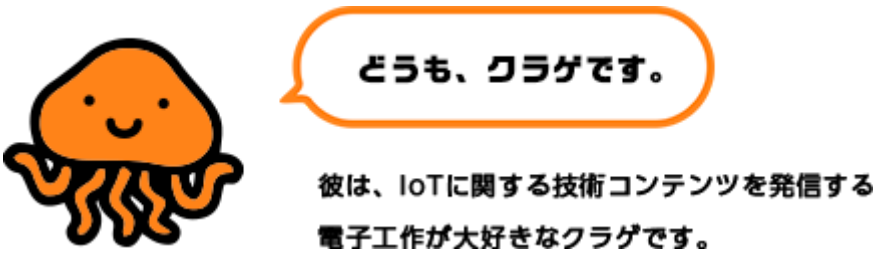
JELLYWARE

MENU

2019/11/27 update

OpenVINO™で始めるディープラーニング



どうも、ディープなクラゲです。

今回から「OpenVINO™ でゼロから学ぶディープラーニング推論」シリーズを掲載してゆきます！

このシリーズは、ディープラーニング概要、OpenVINO™ツールキット、Neural Compute Stick、RaspberryPiの使い方、Pythonプログラミングをゼロから徹底的に学び、成果としてディープラーニング推論アプリケーションが理解して作れるようになることを目指します。

今回は、どのような環境が必要なのか？ ディープラーニングとは？ OpenVINO™ツールキットとは？ について説明します！

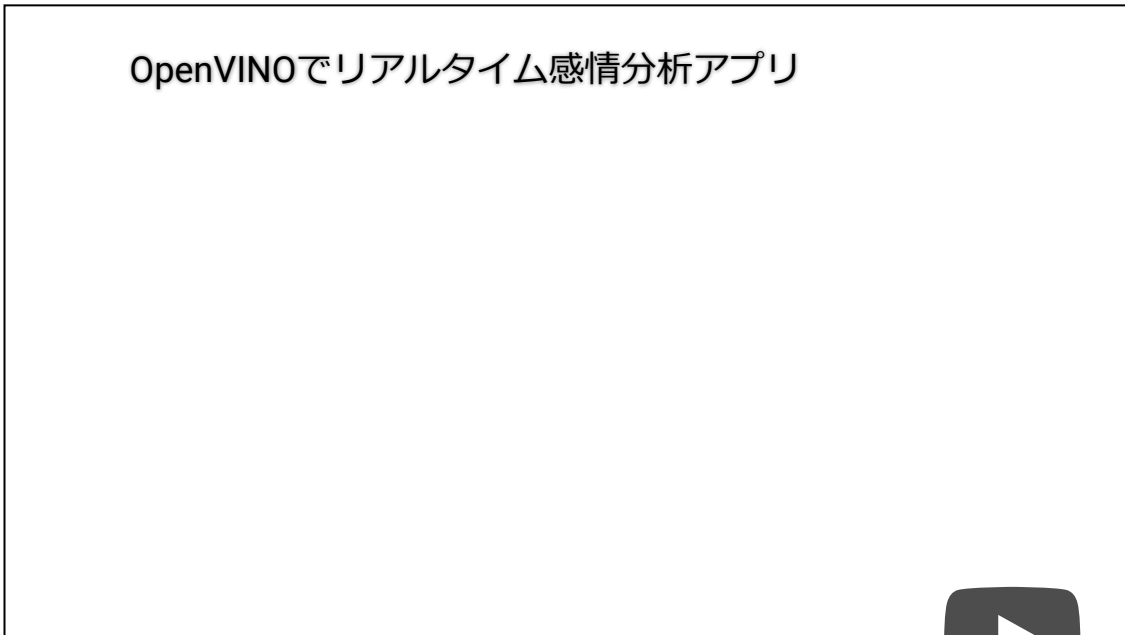
【 目次 】

何が出来るようになるのか？

本シリーズでは全９回（予定）の記事を通して、Neural Compute StickとOpenVINO™ツールキットを使ったディープラーニング推論のプログラミングについて、徹底的に詳しく解説します。

さて、気になるのはこのシリーズに付き合うと一体何ができるのか？ということだと思います。
「結局、mnistやって終わりかよ！」では意味がありません。
※ちなみにmnist(エムニスト)とはディープラーニングのHello, worldです。

こちらの動画を見て下さい



動画に映っているアプリと同じものが理解して作れるようになります。
主に以下の３つのステップで順にアプリを作り上げます。

- 顔画像から感情をディープラーニングで分類
- カメラ映像から顔をディープラーニングで検出
- 顔検出と感情分析を組合せたグラフィック表示を行う

サーバーなどは一切使わず、カメラを使ってリアルタイムにディープラーニング推論を行っているのが特徴です。
このようなアプリを作ることにより、ディープラーニングでは何が得られて、どのくらいの精度なのかが身をもって体感できます。書籍や講義では得られない、「ディープラーニングで出来ること／出来ないこと」を感じ取りましょう！

必要な機器について

本シリーズは主に以下のハードウェアを前提としてサンプルコードや解説を作成していますが、実はPCさえあれば出来る場合があります。詳細は後述します。

- PC
- Neural Compute Stick
- Raspberry Pi
- USBカメラ

また、ソフトウェアとしてはOpenVINO™ツールキットを必要としますが、無償でダウンロードしインストールすることが可能です。特に有料版のようなものはなく、動作や機能に制限はありません。
なお、本シリーズの全コンテンツは完全無料で閲覧可能です。つまり、必要なハードウェアさえ揃っていれば、無料ですぐにでもディープラーニング推論できるということです！

必要なスキルについて

本シリーズではディープラーニング推論のプログラミングを行います。最低限、PCの一般的な操作やキーボードでタイピング出来ることが必須となります。これまでに何らかのプログラミング言語を使って初級レベルでコードを作成した経験があることが望ましいですが、初心者の方向けにも丁寧に説明しますので、安心してください。
その他、数学的な知識や、データサイエンス・機械学習の知識、RaspberryPiの使い方などを知らなくても大丈夫です。

ディープラーニングの種類と用途

ディープラーニングの種類として主に以下４つが挙げられます。これらはディープラーニングのネットワークと呼ばれるものです。「ネットワーク」とは「手法」という意味で捉えておけばOKです。

- RNN(Recurrent Neural Network)
- DQN(Deep Q-Network)
- GAN(Generative Adversarial Network)
- CNN(Convolutional Neural Network)

それぞれのネットワークは得意な領域が異なります。特徴を以下に記します。

RNN：自然言語処理

音声認識や外国語の翻訳などを処理します。数年前にGoogole翻訳が飛躍的に向上したのはディープラーニングによるためと言われています。
RNNの進化版としてLSTMというネットワークも有名です。RNNやLSTMは他にも株価予測や売上予測など時系列データの予測にも使われています。

DQN：強化学習

強化学習とは簡単にいうと、赤ちゃんが成長するように学習する技術です。赤ちゃんは限られた環境の中で様々な動きにチャレンジして成功と失敗を繰り返して、だんだんと学習してゆきます。強化学習の代表例はGoogleが開発し、世界ランク一位のプロ囲碁棋士を破ったAlphaGo（アルファ碁）です。クルマやドローンなどの自動運転技術にも使われています。

GAN：画像生成

ここで言う画像生成とは、画像の特徴を捉えて、似たような画像を作り出すものです。
例えば、猫の画像をゴッホ風にしたり、テキストから画像生成を行うということが出来ます。また最近では架空の顔や人物を作り出すことが可能となり、本物の写真か、生成された画像なのかの見分けがつかなくなるくらいに性能が向上しています。

CNN：画像認識

ディープラーニングで最も精度が高いのが画像認識です。画像認識はコンピュータビジョン、つまりロボットの目とも呼ばれています。既に人間の目の精度を超えたとされています。既に工場の目視検査や医療の画像診断などに活用されています。また小売り業や自動運転にも欠かせない技術になりつつあります。
本シリーズでは、この画像認識を取り扱います

CNN画像認識技術

さらにディープラーニング画像認識にも、大きく分けて３つの種類があります

- 画像分類
- 物体検出
- セグメンテーション

それぞれについて具体的に解説します

画像分類

「何」であるかを推測する技術です。
例えば、猫なのか犬なのか人間なのかを判別することができます。

物体検出

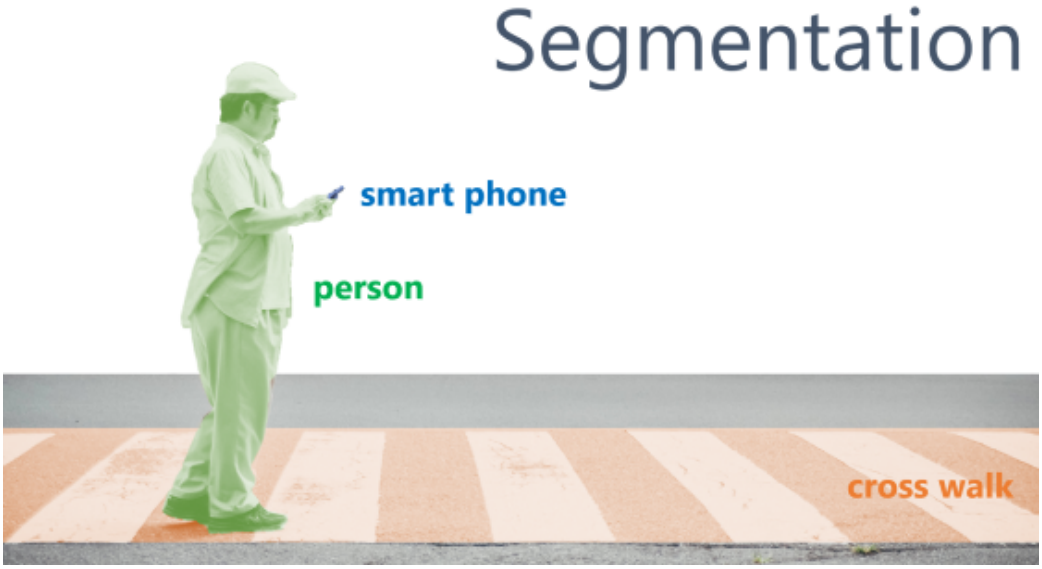
「何」が「どこ」にあるかを推測する技術です。

例えば、人や道路の位置やおおよその領域を検出することができます

セグメンテーション

ピクセル単位で、領域を分けることができる技術です。単に物体検出の精度が上がっただけでなく、「どういう状況なのか」までを把握することができるようになります。主に自動運転に活用されています。

下図はイメージですが、横断歩道の上にスマートフォンを持った人ということが分かれば、「歩きスマホで危険だ」と状況判断できます。



このように画像認識技術は進化していて、「画像分類」→「物体検出」→「セグメンテーション」の順でコンピューターの計算負荷が増え、処理時間も増えることになります。

本シリーズでは「画像分類」と「物体検出」そして両者の組合せを取り

扱います。なお「セグメンテーション」については、OpenVINO™ツールキットのサンプルモデルにもありますので、本シリーズを修了した後に、そちらを実行してみることをお勧めします。

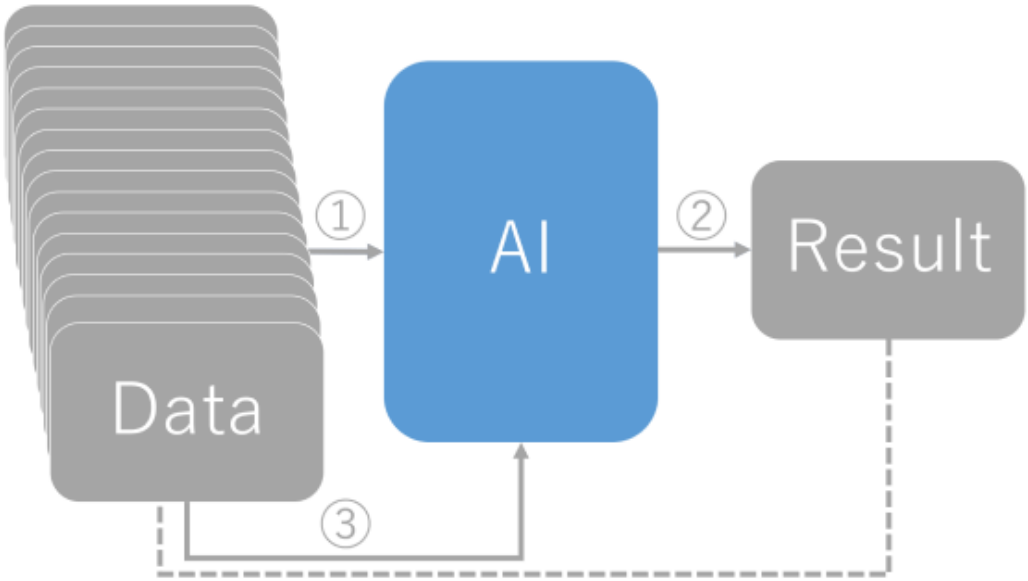
学習フェーズと推論フェーズ

ディープラーニングのフェーズは主に２つに分かれます

- 学習フェーズ
- 推論フェーズ

学習フェーズ

「学習」は英単語だと'learning'ですが、'trainning'という単語も良く使われます。
AIはある程度テンプレートのようなものがあり、初めに人間がどれを使うかを選択します。しかし、最初の時点では全く学習できていないため、データに対してデタラメな予測しかできません。
AIに大量のデータを与え予測させ、正解値との誤差を出し、誤差が小さくなるようにAIにフィードバックして調整します。これを繰り返すことにより、最初はデタラメだった予測値がだんだんと正解値に近づいてゆきます。
イメージですが、図で描くとこのような感じです

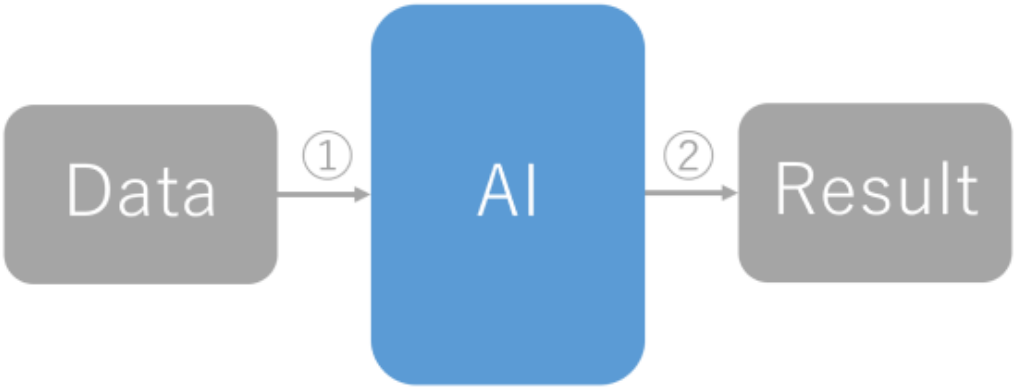


1. 大量のデータをAIに渡して計算させる
2. AIからの予測結果を受け取る
3. 予測値と正解値の誤差が小さくなるようにAIパラメータを調整する

１度で終わりではなく、誤差が収束するまで何度も繰り返します。
学習フェーズでは、予測結果を出して終わりではなく、「大量データ」「誤差をフィードバックして改善」「何度も繰り返す」という項目があるため、非常に処理時間がかかります。

推論フェーズ

「推論」とは予測するという意味に近いです。英単語だと 'inference' や 'predict' などが良く使われます。
推論フェーズは非常にシンプルです。学習フェーズで完成したAIに対し、データを入力し、予測結果をもらうだけです。



1. データをAIに渡して計算させる
2. AIからの予測結果を受け取る

左から右へ１回流せば完了です。
「大量データ」「誤差をフィードバックして改善」「何度も繰り返す」という項目はないため、処理時間が軽いことが分かると思います。

AIの正体はモデルと重み

AIの中身は何が入っているのでしょうか？

よく「人間の脳の神経細胞（ニューロン）の仕組みを模した構造」などと説明されますが、一切忘れて下さい。この表現だと、AIがとても高度で崇高な技術であると錯覚してしまいます。

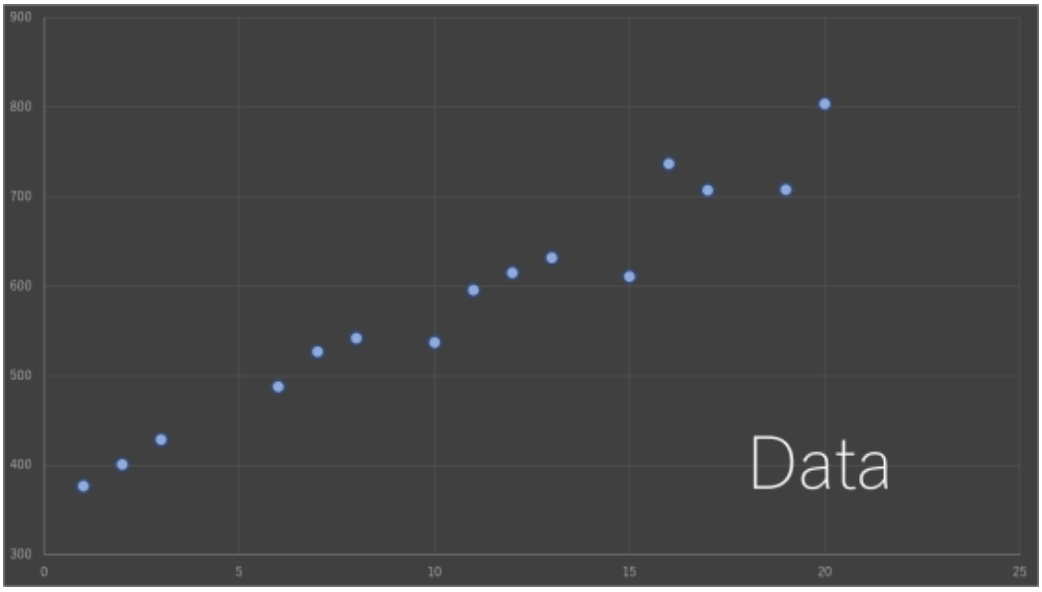
AIはただコンピュータがひたすら計算を行っているだけです。AIに必要な計算機には、生物的なICや脳の構造のような回路が入っている訳でもありません。計算の処理の仕方が脳の伝達っぽいというだけのことです。皆さんがお持ちの普通のパソコンやスマホでもAIは出来ます。

（ただし、ひたすら計算を行うため、コンピュータのスペックが低いとひたすら時間がかかってしまいます）

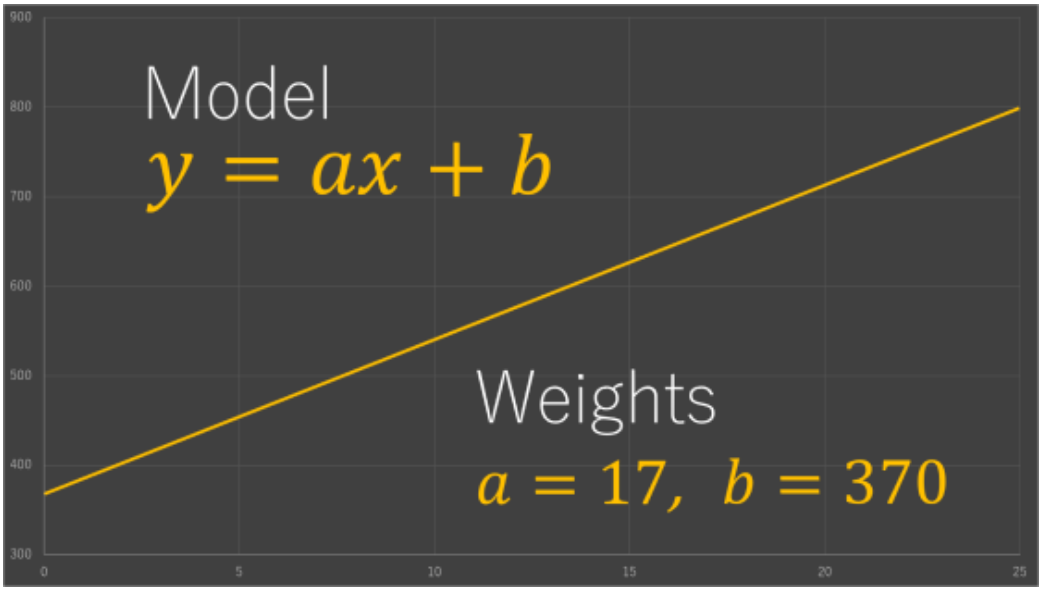
さて、ひたすら計算を行っているだけのAIですが、中身を大きく２つに分けると「モデル」と「重み」に分かれます

これだけだと何だか分からないと思いますので、イメージで説明します。

AIを学習させるにはデータが必要です。例えば、横軸が勤続年数で縦軸が年収というデータをグラフに表示したとします。

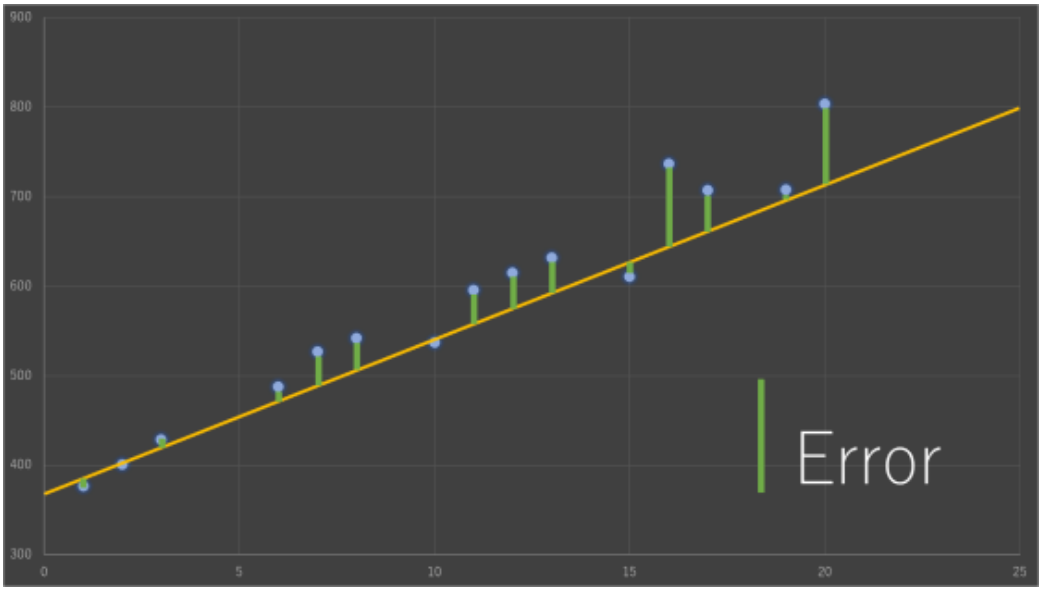


最初は人間がテンプレートを選びます。例えば `y = ax + b` という一次関数の直線を採用したとします。この直線そのものがモデルで、直線の傾き `a` と切片 `b` が重みです。



先程のデータと直線の誤差が緑の棒です。最初の３点はほぼ合ってますが、中盤から後半にかけては結構誤差があります。これが小さくなるよ

うに重み`a`と`b`を調整するのが学習フェーズです。



この例では、「勤続年数」という１つの次元しかありませんでしたが、実際には「業界」「職種」「性別」・・・など数多くの次元が必要でしょう。そして次元が大きくなると１次関数や２次関数といった簡単なモデルでは表現できなくなり、ディープなモデルが必要になります。また大量のデータも必要になってきます。

簡易的なイメージでAIの中身について説明しましたが、「AIとはモデルと重みである」ということが理解できればOKです。なお、学習が完了した「モデルと重み」のことを「学習済みモデル」と呼びます。

OpenVINO™ツールキットとは

公式ページを見ると色々と書いてありますが、クラゲ視点でわかりやすく初心者向けにまとめます。

OpenVINO™

無償のソフトウェア

無料でダウンロードとインストールができるだけでなく、全機能を制限なく使うことができます。期間限定、有料版、広告表示などもありません。商用利用も含めて無償です。

2018年9月にリリースされ、何度かバージョンアップされており、安定しています。

エッジコンピューティング

ディープラーニングの実行環境として「クラウドコンピューティング」と「エッジコンピューティング」の２つに大きく分かれます。
「クラウドコンピューティング」はクラウド側でディープラーニングを実行することを指します。ある程度までは無料で使える場合がありますが、期間やデータ上限などを超えると有料になるケースがほとんどです。また、カメラで撮った写真をアップロードして推論することは出来ますが、カメラ映像をリアルタイムに推論することは困難です。
OpenVINO™ツールキットはクラウドを使用しない「エッジコンピューティング」です。カメラ映像をリアルタイムに推論することも可能ですし、データをネット上にアップしないので、通信料ゼロでセキュリティ的にも安心感があります。

何が入っているのか？

OpenVINO™ツールキットには以下のツールやライブラリ、サンプルが入っています。

- Inference Engine(IE)
- Model Optimizer(MO)
- OpenCV
- OpenVX
- Sample Applications

Inference Engineは、推論エンジンのことで、プログラミングで呼び出して使います。
Model Optimizerはモデル変換ツールです。後ほど説明します。
RaspberryPiのOpenVINO™ツールキットにはModel Optimizerは含まれません。
OpenCV, OpenVXはコンピュータビジョンライブラリです。本シリーズではOpenCVを使います。OpenCVはカメラ映像の入出力、文字や図形

描画、ちょっとした画像処理などに使います。RaspberryPiのOpenVINO™ツールキットにはOpenVXは含まれません。

Sample Applicationsは、サンプルアプリやソースコードです。非常にたくさんあります。

OpenVINO™ツールキットで出来ること

ディープラーニング推論

OpenVINO™ツールキットでは「ディープラーニング推論」を取り扱います。

言い換えると「ディープラーニング学習」は出来ないということです。学習させるにはディープラーニング用の「フレームワーク」を使って行う必要がありますが、学習は必須ではありません。実は、ディープラーニングに関わる技術は様々なものが無料で公開されています。フレームワークもそうですが、画像を処理するためのライブラリ、学習するための大量データ、モデル、そして「学習済みモデル」もたくさん公開されています。本サイトでは豊富にある「学習済みモデル」を活用します。

CNN画像認識

ディープラーニングには「RNN」「DQN」「GAN」「CNN」などがあります。OpenVINO™ツールキットで主に対応しているのは「CNN」です。

「モデル」は様々な「レイヤー」というもので構成されており、OpenVINO™ツールキットのバージョンアップと共に対応レイヤーが増えています。またプロセッサによっても対応レイヤーが異なるので注意してください。詳細は以下のウェブサイトを参照して下さい。

Supported Framework Layers

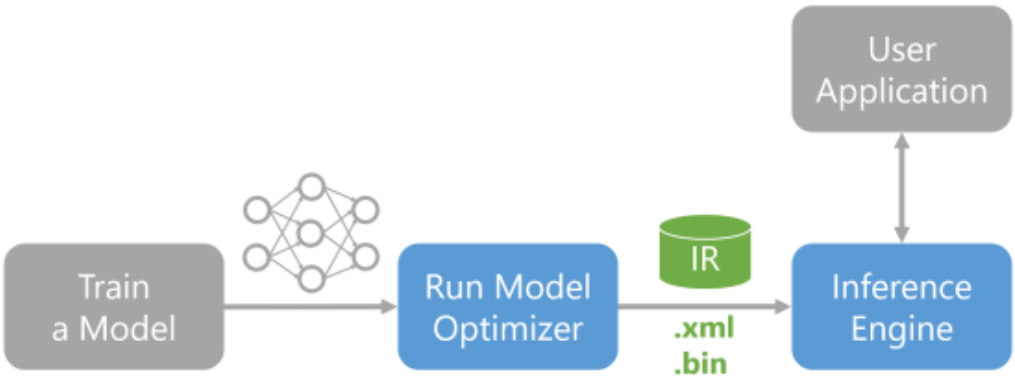
Processor vs Supported Layers

学習済みモデルの変換

OpenVINO™で推論実行可能な「学習済みモデル」はIR(Intermediate Representation)という形式のみです。IRは２つのファイルで、それぞれ

`xml`と`bin`の拡張子です。

他の学習済みモデルの形式の場合は、OpenVINO™ツールキットに入っているModel Optimizerを使ってIRに変換します。



Model Optimizerが対応している「学習済みモデル」の形式は以下の通りです。

- Caffe形式(.caffemodel + .prototxt)
- TensorFlow形式(frozen.pb, non_frozen.pb + .ckpt)
- MXNet形式(.param)
- ONNX形式(.onnx)

フレームワークはPyTorch, Chainer, NNC(Neural Network Console)など、他にも多数ありますが、ディープラーニングの共通フォーマットである「ONNX」形式で出力することにより、OpenVINO™で扱うことが可能です。

OpenVINO™ツールキットの対応環境

OS

対応OSです。

- Linux
- Windows10
- MacOS

対応OSバージョンと対応プロセッサもご確認ください。詳細はGET STARTEDの各OSに対する要求システムが書いてある箇所をチェックし

て下さい。なお、RasapberryPiで開発する場合はOSを気にする必要はありません。
プロセッサについては以下で説明します。

プロセッサ

プロセッサとは主に以下のようなものを指します。

- CPU
- GPU
- FPGA
- VPU

CPU, GPU, FPGAは一般的な名称なので説明は割愛しますが、VPUだけ軽く説明すると、"Vision Processing Unit"の略で、現状だと"Neural Compute Stick"と"Neural Compute Stick2"の２つのことを指します。なお、`VPU` ではなく `MYRIAD` という表現を使う場合もあります。Myriad™は中に入っているチップの名称です。
OpenVINO™ツールキットは様々なプロセッサに対応していますが、インテル製の比較的新しいものに限られますので注意して下さい。インテルCPUには Xeon, Core, Pentium, Celeron, Atomなどがあります。
ちなみに、"Development Environment Installation Guides"には、CeleronやAtomの記述がないですが、LinuxOSでApollo Lake世代以降であれば動作できるようです。
なお、RasapberryPiで開発する場合はプロセッサを気にする必要はありません。

Raspberry Pi

Raspberry PiのCPUはARMであり、インテル製ではありません。
そのままではOpenVINO™のInference Engineを動作させることができませんが、Neural Compute Stickを挿して使うことにより動作可能になります。
なお、Raspberry PiのOS "Raspbian" はOpenVINO™ツールキットに対応しています。

本サイトにおけるOpenVINO™開発方針

OpenVINO™を使った開発手法は、たくさんのバリエーションがあるため、初心者は迷ってしまいます。そこでグラゲがバシッと開発方針を決めます！

推奨するOpenVINO™開発環境

OpenVINO™ツールキットを使う開発環境として、主に以下の３通りが考えられます。

- 環境1 : PC のみ
- 環境2 : PC + Neural Compute Stick
- 環境3 : PC + Raspberry Pi + Neural Compute Stick

環境1に関しては、お持ちのPCが先程の対応環境に該当していることを確認してください。
環境2に関しては、お持ちのPCのCPUがx86系64bitでOSがUbuntu16.04もしくはWindows10が対象です。詳細はこちらをご確認ください
環境3に関しては、お持ちのPCのスペックは問いません（SDカードにデータ書き込みできればOK）

Raspberry Piであれば、気兼ねなくインストールできることと、実証実験しやすいというメリットがあるため、本サイトでは 環境3 の **"PC + RaspberryPi + Neural Compute Stick"という開発環境をベースに解説**を進めてゆきます。 具体的にどこで何を購入すれば良いかは次回説明します。

※ただし、環境2の場合だとしても本サイトのコードそのままでいけると思いますし、環境1の場合だとしても少しだけコード変更すればいけるかと思います。

OpenVINO™における推論手法

OpenVINO™ツールキットでディープラーニング推論を行う方法として３つあります

1. 自分でオリジナルなモデルを学習させてから推論
2. パブリックな学習済みモデルを使った推論
3. インテルの学習済みモデルを使った推論

推論手法	フレームワークで学習	Model Optimizerで変換
1. Original Model	必要	必要
2. Public Model	-	必要
3. Pre-Trained Model	-	-

1はTensorflowやCaffeなどのフレームワークを使って事前に学習させる必要があり、初心者にとっては非常に学習コストが高いです。2や3でディープラーニングに慣れてからこちらに進むことを推奨します。

2のパブリックな学習済みモデルは、一般的に公開されている学習済みモデルのことです。例えば、GoogLeNet、SqueezeNet、SSDやYoloなどです。詳細はこちらを参照して下さい。
1と2はModel Optimizerを使っての変換が必要となります。3で推論に慣れてから行うと良いと思います。

3のインテル学習済みモデルは、いきなりプログラミングで取り込んで使うことができます。
人の姿、顔のパーツ、クルマ、バイク、歩行者などを対象とした、実用的な学習済みモデルがたくさん用意されています。ジャンルとしては、Retail系(小売りや屋内向け)、Adas系(自動運転向け)、Barrier系(セキュリティカメラ向け)などがあります。詳細はこちらを参照してください。

結論として、本サイトでは 3の**インテルの学習済みモデルを使った推論**を使って解説します。

OpenVINO™でのプログラミング手法

OpenVINO™ツールキットでプログラミングを行う手法として 3 つ挙げます

1. C++
2. Python
3. OpenCVのDNNを使ったPython

推論はOpenVINO™ツールキットのInference Engineを呼び出して使いますが、1,2はそのまま呼び出す方法で、3はOpenCVを経由して呼び出す方法です。"OpenCVのDNNを使ったC++"も可能ですが、ほぼメリットがないので割愛します。
1,2は推論エンジン呼び出しにはOpenCVを使いませんが、それ以外の処理ではOpenCVを活用します。

それぞれの特徴を表にまとめました

プログラミング手法	機能・性能	コードのシンプルさ	実行のお手
1. C++	◎高い	△やや複雑	△ビルド
2. Python	○	○	○
3. Python with OpenCV DNN	×制限あり	○+	○

1は中級者以上の方向けです。普段からC++を使いこなしている方はこれがベストです。
2は最も安定しており、デメリットがありません。
3は「×制限あり」のデメリットがあります。具体的には、非同期推論が出来なかったり、まだバグがある点です。これは今後のOpenCVのバージョンアップに期待したいと思います。

結論として、本サイトでは2の **Pythonでプログラミング**を行います。

(Appendix) 便利なリンク

OpenVINO™ツールキットの情報が非常に多く、迷ってしまうため、クラゲが良く見るページのリンクを貼っておきます。後で迷ったときに活用してください。

トップページ

- Intel® Distribution of OpenVINO™ toolkit
- Intel® Neural Compute Stick

インストール関連

- RaspberryPi + Neural Compute Stick 環境へのOpenVINO™インストール手順
- PC + Neural Compute Stick 環境へのOpenVINO™インストール手順
- PCのみ 環境へのOpenVINO™インストール手順

インテル学習済みモデル関連

- Pre-Trainde ModelのIRデータ(OpenModelZoo)
- Pre-Trained Modelの解説
- Pre-Trained Modelに対するCPU, GPU, VPUなどのサポート表

その他情報

- OpenVINO™がサポートしているフレームワークのレイヤー
- Neural Compute Stickがサポートしているネットワーク
- YouTube Intel OpenVINO チャンネル
- OpenCV4.1 dnn moduleのリファレンス
- JellyWareクラゲのIoTイベント

以上、「OpenVINO™で始めるディープラーニング」でした。


Neural Compute Stick と
OpenVINO™ でゼロから学ぶ
ディープラーニング推論

Newsletter

ご登録いただくと、新商品やイベント情報をお知らせします。

メールアドレスを入力してください 登録

個人情報について

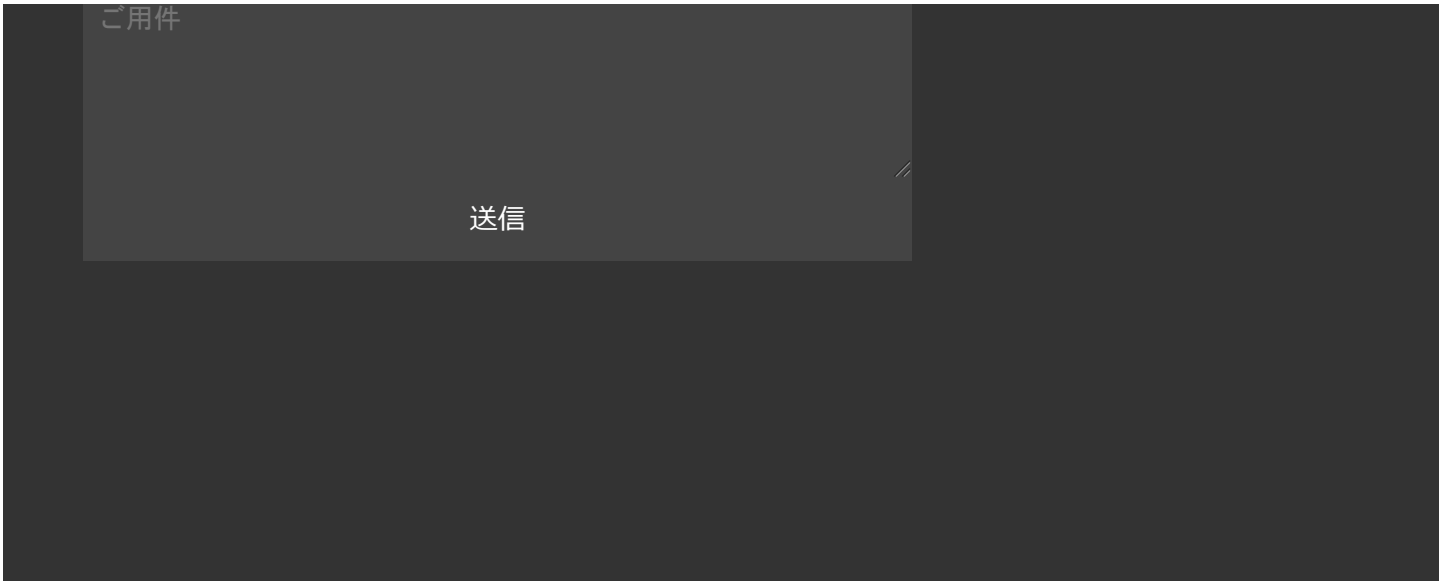
JELLYWARE

〒160-0004 東京都新宿区四谷2-3-6 パルム四谷702号室
03-6273-0758
info@jellyware.jp

個人情報について

CONTACT

お名前
メールアドレス
メールアドレス確認
お電話番号（任意）
ご件名


ご用件
送信