第3章 サイズ変更/チャネル削減/モデル構造変更

ラズパイ検出速度向上… 学習モデルの改良にトライ

奥村 義和

トライすること

● 自分で学習モデルをチューンアップ

物体検出をラズベリー・パイなどの安価なデバイスで推論できると有用です．tiny YOLOなど軽量モデルも提案されていますが，非力なエッジ・デバイスにはまだまだ大きいというケースもあります．自分で学習モデルをチューンアップできると，選択肢が広がります．

● 処理速度…30fps

本稿では，モデルを調整することでラズベリー・パイ4でリアルタイム物体検出を目指します．ここでリアルタイムとは，人間の目でスムーズに動画が再生されていると感じる30fps（Frames Per Second）を目指します．1秒間に30枚なので，画像1枚当たり33msが目標になります．

● 精度…val mAP 16.6%

精度はYOLOv3-tiny(1)相当を目指します．具体的にはCOCO注1データセットでval mAP 16.6%とします注2．推論フレームワークは利用しますが，モデル調整が目的なので量子化は利用しません．以下本稿で記述するmAPは，COCOデータセットでのval mAPを指します．

● モデルのチューンアップ手順

モデルのチューンアップはOSS（Open Source Software）の物体検出モデルで行います．設定ファイルで可能な範囲としますが，次の順に進めます．

1，モデルの再学習を伴わずにできること（入力サイズの調整）
2，モデルの微調整（チャネルの調整）
3，モデル構造の変更（neckの構造変更）

使うミドルウェア

● 推論…ONNX Runtime

ラズベリー・パイ4における推論のフレームワークとしてONNX Runtimeを利用します．

`https://onnxruntime.ai`

推論フレームワークは，その名の通り推論に特化して作成されたフレームワークです．式レベルの計算の最適化や，ハードウェア的な最適化を行うことで，高速な推論が可能です．また，推論フレームワークを使うことによって学習時とは異なるOS，異なる言語を利用できます．

ONNX Runtimeは，ONNX（`https://onnx.ai`）という機械学習モデルのオープン・フォーマットに対応した推論フレームワークです．Execution Providerという仕組みでCPU/GPU/FPGAなどのハードウェア・アクセラレータを切り替えることができます．PCやスマートフォンなどの多様な環境や，Pythonだけでなく C++/C# などの複数の開発言語を利用可能です．

今回は，想定する学習フレームワークがONNX化をサポートしていることから，ONNX Runtimeを利用しました．

なお，他の推論フレームワークとしてTensorFlow Liteなどがあります．ハードウェア・メーカ主導のものもありTensorRT（エヌビディア）やOpenVINO（インテル）などがあります．

● 学習…PyTorchで実装されたNanoDet

学習のフレームワークとしてはPyTorchを利用します．さらにPyTorchで実装されたNanoDet（`https://github.com/RangiLyu/nanodet`）というOSSを利用してモデルを学習します．

注1：`https://cocodataset.org/#home`，COCOは 物体検出でよく使われる有名なデータセットです．33万枚，80オブジェクト・カテゴリ（クラス）からなります．

注2：val mAPは，COCOの検証用データに対する評価指標の1つです．予測した矩形の正答率に関する指標で，高いほど良いです．詳しくは下記URLをご覧ください．
`https://cocodataset.org/#detection-eval`