PS2-19

ONNX2MPSNNGraph: モバイル深層学習コードジェネレータの実装と評価

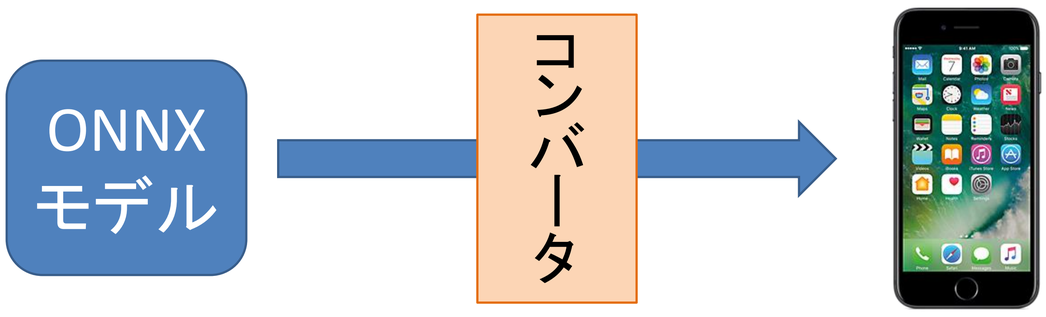
泉 裕貴，柳井 啓司 (電気通信大学)

はじめに

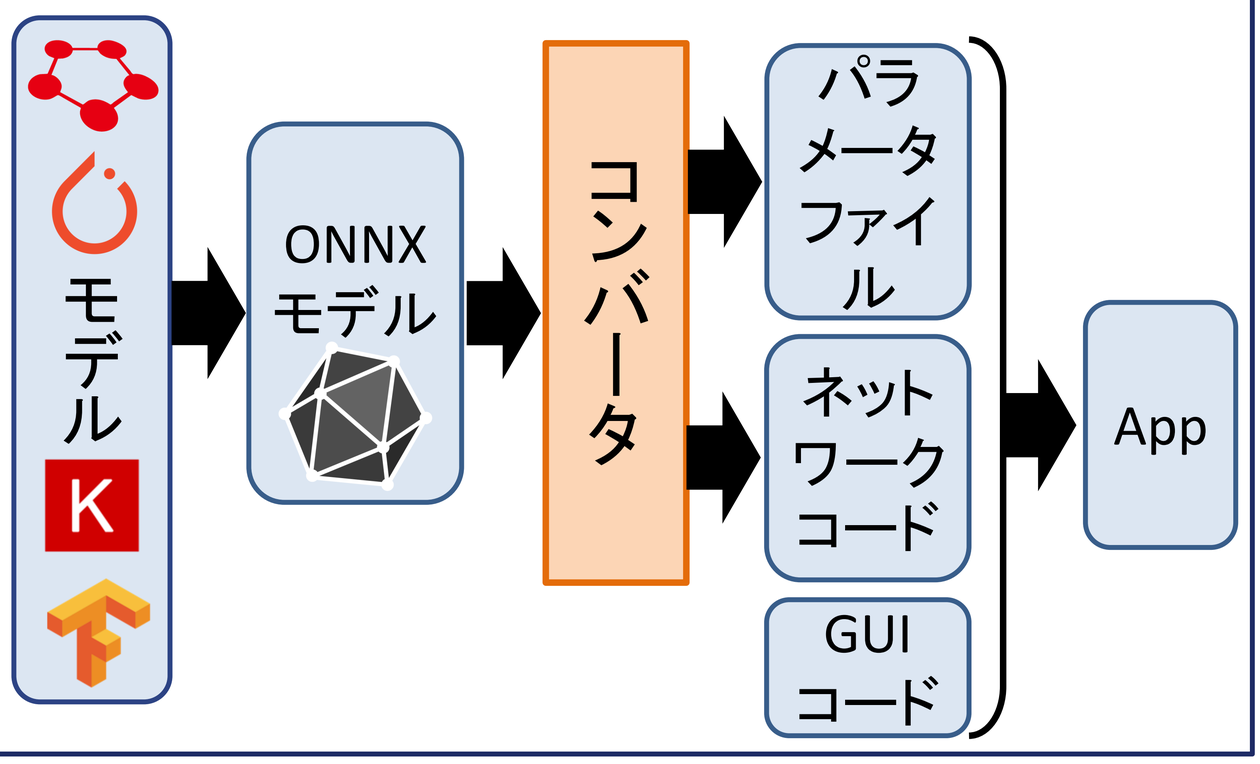
モバイル端末の性能が向上したことで端末上での深層学習の実行が可能となってきた
しかし…

- 実装がブラックボックスで修正が困難【CoreML】
- 特定のフレームワークからの実装のみ対応
  【Chainer2MPSGraph（昨年発表）】

複数フレームワークに対応した修正可能なコンバータを作成



アプリ作成のワークフロー



ONNX2MPSNNGraph

ONNX

深層学習のモデルを現すためのオープンフォーマット

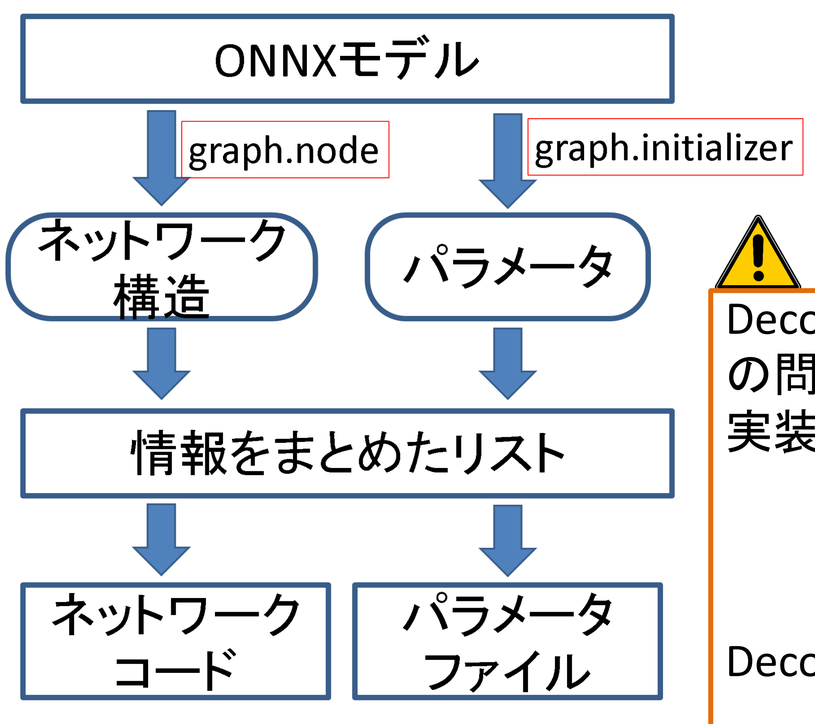
PyTorch Chainer Caffe2 K ONNX

MPSNNGraph

Metal Performance Shaders (MPS)の中のライブラリの1つ

- iPhoneに搭載されているGPUの利用
- グラフ構造の利用

ONNX2MPSNNGraph



DeconvolutionはMPS側の問題でiOS12以降では実装できない

Deconv → UpSampling ＋ Conv

で対応可能

- 生成コード例

```
let conv2 = MPSCNNConvolutionNode(source:pool1.resultImage,
                                  weights: DataSource3("conv2", 1, 1, 64, 64, 1,
                                                       "bn2", 64, useBias: true))
let relu2 = MPSCNNNeuronReLUNode(source: conv2.resultImage)
let conv3 = MPSCNNConvolutionNode(source:relu2.resultImage,
                                  weights: DataSource3("conv3", 3, 3, 64, 64, 1,
                                                       "bn3", 64, useBias: false))
let relu3 = MPSCNNNeuronReLUNode(source: conv3.resultImage)
let conv4 = MPSCNNConvolutionNode(source:relu3.resultImage,
                                  weights: DataSource3("conv4", 1, 1, 64, 256, 1,
                                                       "bn4", 256, useBias: true))
```

実験

ResNet-50で認識速度を比較 [msec]

実装方法	MPSNNGraph	CoreML
iPhone8	86.02	89.12
iPhone8Plus	86.06	89.18
iPhoneXs Max	81.22	23.90
iPad Pro 11inch(2018)	40.94	19.29

結果

- iPhone8Plus以前の端末 → MPSNNGraphの方が高速
- iPhoneXs Max以降の端末 → CoreMLの方が高速

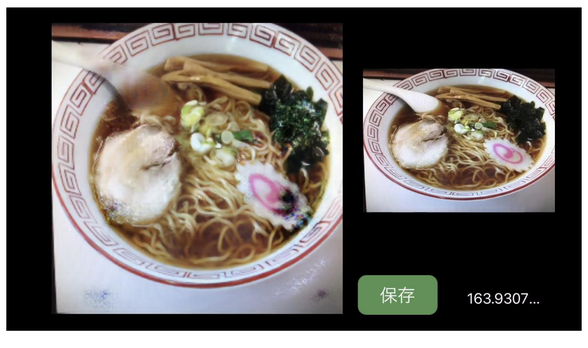
→ Neural Engineの利用の差

サンプルアプリ

iPad

ResNet-50

レンゲ消し



まとめ

Neural Engineを利用するAPIは公開されていない
→ 現状では、CoreML実装の方が高速
APIが公開されればMPSNNGraph実装の方が高速になる可能性がある
→ 作成したコンバータの有用性が見込める