C-013

姿勢推定モデル「MoveNet」における高速化の手法の提案
Proposal of a speed-up method in the posture estimation model "MoveNet”

田中 康雅‡　　中西 知嘉子‡
Yasumasa Tanaka　Chikako Nakanishi

1. はじめに

近年，姿勢の改善や不審な人物の検知の必要性から姿勢推定と呼ばれる技術が注目されている．姿勢推定[1]とは，人間の写った静止画や動きのある動画に対して，関節点を推定しその関節点を線で結ぶことにより人間の姿勢を検出する AI である．その AI のモデルとして，人間の関節点を学習し動画や静止画からリアルタイムに関節点を結ぶことにより人間の姿勢を検出するモデルを姿勢推定モデル[2]と呼ぶ.

この姿勢推定技術は，現在スポーツの分野や医療現場，監視カメラなどで使用されている．そこで，姿勢推定を小型でネットワークを用いず，手元のエッジ端末上で AI 技術を動作させることのできるエッジ端末で動かすことができれば，精密機械のある医療現場などでも扱いやすく，監視カメラなどにも組み込みやすくなるのではないかと考えた．しかし，リアルタイム性が要求されるような応用では姿勢推定モデルは演算量が多すぎるため，性能の低いエッジ端末では処理ができない問題点がある．

そこで本研究では，CPU と FPGA が同一チップ上に存在する SoC FPGA ボードである Ultra96v2 を使用する．そして，専用の回路を作成することにより低リソースで推論処理を行いつつ，ソフトと回路を協調動作させることによって更なる高速化を図る．それにより，エッジ端末上でリアルタイムに実行を目指すことを目的とする．

2. 使用機材・使用ネットワーク

2.1 Ultra96v2

Ultra96v2[3]とは Avnet 社より開発された SoC FPGA ボードである．SoC FPGA ボードとは CPU と FPGA が同一チップ上に存在するものである．このボードを使用する利点として，複雑な処理のアルゴリズムは CPU 処理で実現し，高負荷な処理は FPGA による回路を実装することにより，高速化を図ることができる点がある．

2.2 MoveNet

本研究で実装する姿勢推定モデルには，2021 年に Google が公開したボトムアップ型の姿勢推定モデルである MoveNet[4]を用いた．MoveNet は 17 個のキーポイントを高速かつ高精度に検出することができるモデルである．また，MoveNet には精度に特化させたモデルである Thunder と速度に特化させたモデルである Lightning の二つのモデルがある．精度に特化させた Thunder は演算量が大きすぎ，ultra96v2 で動かすには適していないため，今回は速度に特化したモデルである Lightning を使う．

2.3 Ceras

Ceras とは先行研究[5]で開発された ONNX 深層学習モデルの構造，重みをテキストデータに抽出し，C++のプログラムで読み込み推論することができるライブラリである．我々は，回路を C++言語を用いて高位合成で作成し，協調動作では CPU と回路で細かく処理を分担する手法をとる．そのため C++言語で推論処理全体を行う方が開発効率が良い．したがって今回は Ultra96v2 上で推論を行うライブラリとして，Ceras を用いた.

2.4 ONNX

ONNX[6]とは Open Neural Network eXchange の略で，ディープラーニングや機械学習モデルのような人工知能モデルを表現するためのフォーマットである．ONNX を使えば異なるフレームワーク間で学習したモデルを使用することができる．2.3 節で説明した Ceras を使用するためには，この ONNX 深層学習モデルである必要がある．MoveNet は Pytorch 言語で公開されているため MoveNet を ONNX 深層学習モデルに変換した．さらに，ONNX のツールである ONNX-Simplifier[7]を使用し，ONNX のネットワーク構造を最適化した.

3. 実装手順

3.1 Movenet 推論時間

初めに MoveNet のプログラムを，Ultra96v2 の CPU のみを使用し全体の処理時間とそれぞれのネットワーク構造の層の処理時間を出力した．その時間を表 3.1.1 に示す.

図 3.1.1　MoveNet の各層の処理時間

	合計時間(ms)	割合(%)
Conv2D	15237.1	95.1
DW_Conv2D	437.748	2.7
Activation	328.96557	2.1
その他	19.6179	0.1
合計時間	16023.43147	

表 3.1.1 から，全体の処理時間は 16023[ms]であることがわかる．また，処理に一番時間のかかっている層は Conv2D 層で 15237[ms]で，全体の約 95%を占めていることがわかった．したがって，この Conv2D 層を回路化することにより高速化を図ることにした．また，Activation 層である ReLU 層や Resize 層も回路化することにした.

回路化するにあたって，Conv2D 層は，先行研究[8]で作成された汎用的な Conv2D 層用の回路を使用する．

3.2 Activation 層の組込み

Conv2D 層の後にある Activation 層を回路に組み込み，回路で実行される Activation 層をモデルから削除することによって全体の層数を削減し高速化を図った．MoveNet の Conv2D 層に続く Activation 層は ReLU 層と Clip 層である．先行研究の回路ではすでに ReLU 層は組み込まれていたので，Clip 層を回路に追加実装した．Clip 層は，ReLU 層の処理に最大値を設定できる層である．MoveNet では最大値は固定値であるため，回路内に最大値を保持するようにした．

3.3 データ転送のオフセット化

先行研究[9]では，畳み込み層の処理に必要な重み情報などを回路実行時に共有メモリに書き込んで DMA 転送によって共有メモリから回路に送っていた．そのため，この書き込む処理に全体の処理時間の約 23%を占めていた．そこで先行研究[9]で作成されたオフセット化という手法を使用することにした．本手法では，推論実行時に変化しない重みなどの情報をあらかじめ共有メモリに保持しておく．各層の重みデータの開始アドレスとサイズはソフトウェアで管理し，実行時の DMA 転送の設定に用いる．この手法により，重みやバイアスを実行時に共有メモリに書き込む時間を削減する．

3.4 共有メモリの活用

現状のプログラムでは回路側で演算した結果は DMA 転送により共有メモリに転送されるが，その後，値を次の層に送るために共有メモリからソフト側のメモリに格納していた．そのため，共有メモリからの読み出し時間が全体の約 14%を占めていた．しかし，次の層が回路で実行する処理であった場合はソフト側のメモリに入れる必要がない．そこで次の層が回路で処理をする層であった場合は，共有メモリを読み出しソフト側のメモリに書き込む処理をなくした．その時，前の層の出力を書き込んだ共有メモリのアドレスとそのサイズは，ソフトウェアで管理するようにした．

4. 検証方法と結果

4.1 検証方法

検証方法としては，MoveNet を Ultra96v2 上で CPU のみで推論をさせたときと，CPU と回路を強調動作させて推論したときの実行時間を計測することで，高速化の検証を行う．

4.2 結果

MoveNet を回路で処理させた結果を図 4.2.1 に示す．

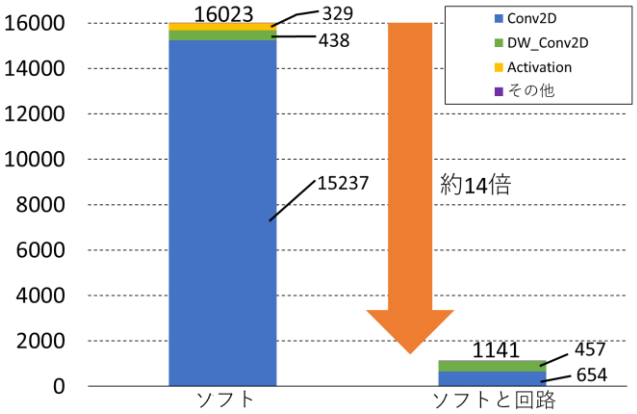

図 4.2.1　実行時間の比較

図 4.2.1 を見ると処理全体の時間がソフトでは 16023[ms]であったのに対し，ソフトと回路で強調動作をさせることにより 1141[ms]という結果となり，約 14 倍高速に推論することができた．また，Activation 層は 3.2 節で説明した通り，Conv2D 層に組み込まれているのでほとんどなくなった．

5. 結論

今回 Conv2D 層と Activation 層を回路化し，ソフト側で回路に適した処理を行うことにより，精度を落とすことなく，推論時間を約 14 倍まで減らすことができた．現状ではいまだに Conv2D 層に一番時間がかかっている状態ではあるが，DW_Conv2d 層の処理時間の占める割合が高くなってきている．

今後の展望としては，DW_Conv2D 層が全体の約 40%を占めているので，DW_Conv2D 層を実行可能な回路を作成することで更なる高徳化ができると考えている．また，層の順番が Conv2D 層，DW_Conv2D 層，Conv2D 層の順となっている箇所が多いためを，DW_Conv2D 層の回路化によって，共有メモリの有効活用が可能になると考えている．

参考文献

[1]]https://otafuku-lab.co/posture-analysis/
[2] https://www.macnica.co.jp/business/ai/blog/142044/
[3] https://japan.xilinx.com/products/boards-and-kits/1-vad4rl.html
[4] MoveNet：動きの激しい動画向け骨格検出モデル - axinc - Medium
[5] 西岡駿, 中西知嘉子, “機械学習ライブラリの C 言語化の実現”, ” 一般社団法人電子情報通信学会” (2021).
[6] https://qiita.com/motoJinC25/items/d662be70b6b9b8ebbaea
[7] https://cyberagent.ai/blog/tech/17300/
[8] 大戸彰馬, 中西知嘉子, "推論処理における畳み込み処理の回路化の検討",電子情報通信学会総合大会(2022).
[9] 岩本征弥, 中西知嘉子, ” エッジ端末による推論処理に必要なデータ転送最適化手段の検討，FIT(2022)

† 大阪工業大学 情報科学研究科 情報科学専攻
Graduate School of Information Science and Technology
Osaka Institute of Technology
‡ 大阪工業大学 情報科学部 情報知能学科
Department of Information and Computer Science Osaka Institute of Technology