C-018

エッジ AI における CPU-回路間のデータ転送方法の改善
Improved CPU-Circuit data transfer method for edge AI

村井 稔大‡　　中西 知嘉子‡
Toshihiro Murai　Chikako Nakanishi

1. はじめに

近年，通信遅延やセキュリティの観点から，ネットワークを用いずにエッジデバイス上で AI を動かすエッジ AI に注目が集まっている．しかし，複雑かつ膨大な量の処理を必要とする高精度な AI をリソースや電力などの制約があるエッジデバイス上に実装してもリアルタイムに処理させることは難しい．

そこで，低消費電力で動作するエッジデバイスを採用し，できる限りコストを削減したエッジ AI の高速化を目指す．先行研究[1]では，エッジデバイスとして SoC FPGA と呼ばれる CPU と FPGA が同一チップ上に存在するデバイスを使用し，負荷が大きい箇所を回路で，それ以外の全てを CPU で処理することで AI の推論時間の高速化を行った．

先行研究の回路を用いた推論時間の内，CPU と回路間のデータの共有に 1033ms かかっており，これは推論全体の約 21%の時間を使用している．本論では，この CPU と回路間のデータ共有方法の改善について説明する．

2. 使用 AI モデル・開発環境

本研究では，AI モデルとして，ResNet50[2]を，エッジデバイスとして SoC FPGA である Ultra96-V2[3]を使用した．

2.1 ResNet[2]

ResNet(Residual Network)とは，Microsoft 社の Kaming He 氏らが考案したニューラルネットワークモデルである．ResNet は残差学習を提案したモデルであり，層を深くできないという劣化問題を解決したモデルである．本研究では，精度やモデルサイズのバランスがよい ResNet50 を採用した．

2.2 Ultra96-V2[3]

Ultra96-V2 とは，AVNET 社より発売されている Arm コア内蔵の SoC FPGA ボードである．SoC FPGA とは，CPU と FPGA が同じチップ上に存在するデバイスで，CPU-FPGA 間の高速なデータ転送が可能な上，高効率な専用回路を作成することで FPGA による高並列で高速な処理を行えるデバイスである．先行研究では，CPU で全体の処理を行いつつ，1 部の高負荷な処理を FPGA に実装した回路で行う事で高速化を図った[1].

2.3 開発ツール

2.3.1 回路生成ツール

回路は，C++言語で記述したコードに高位合成を使用して RTL 言語に変換することで作成した．高位合成を行うツールとして Vitis HLS2022.2[4]を，回路生成を行うツールとして Vivado Design Suite[5]を使用した．

2.3.2 Ceras[6]

Ceras は弊研究室で開発されたライブラリで，学習済み ONNX モデルの推論を C++言語で実行するライブラリである．一般的な AI は Python 言語で作られている．しかし，Python での処理はブラックボックスな一面が大きいため，開発の検証が容易になる Ceras を開発に用いた．

2.4 使用回路

今回は回路の改善ではなく，回路を取り巻く処理を改善するため，先行研究[1]と同じ動作をする回路を用いる．回路には保持できるデータ量に限界があるため，この限界を超えない範囲で回路が保持するデータサイズを設定している．表 2.1 は回路のデータサイズの上限一覧である．

表 2.1　回路のデータサイズ上限

カーネルサイズ	データの種類	データサイズ	
		ResNet50	回路の上限
カーネル 1×1	入力サイズ	56	58
	チャネル	2048	576
	カーネル	2048	128
カーネル 3×3	入力サイズ	56	58
	チャネル	512	64
	カーネル	512	128

ResNet50 には入力サイズ 230，カーネルサイズ 7x7，ストライド 2 の層が 1 層だけ存在するが，この層は回路で処理せず，CPU 上で演算を行うため，表 2.1 には記載していない．

3. 提案手法

CPU と回路間でのデータ共有には共有メモリを介して行われる．回路へのデータ転送は，CPU から共有メモリにデータをコピーし，DMA 転送を用いて共有メモリから回路へ行われる．回路で処理を行った結果は，回路から共有メモリに転送され，共有メモリから CPU にコピーする．また，次の層へと処理が移る際にもデータの受け渡しが行われている．図 3.1 は，この従来のデータ転送のイメージを示したものである．

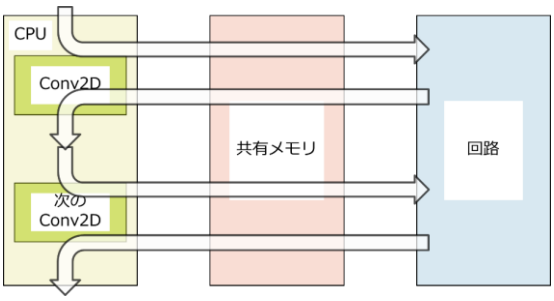

図 3.1　従来の CPU-回路間のデータの流れ

しかし，先行研究[1]では，層を結合し，回路でまとめて処理することで CPU での演算を減らし，回路での演算が連続する場合がかなりの割合を占めるようになった．回路の使用が連続する場合，「共有メモリから CPU へのデータ読み出し」や「CPU から共有メモリへのデータ書き込み」，「層間のデータの受け渡し」は必要ない．そこで，本研究では，これらの処理を省略し，データ共有に要する時間の短縮を行う．図 3.2 は，提案手法を適用した後のデータ転送のイメージを示したものである．

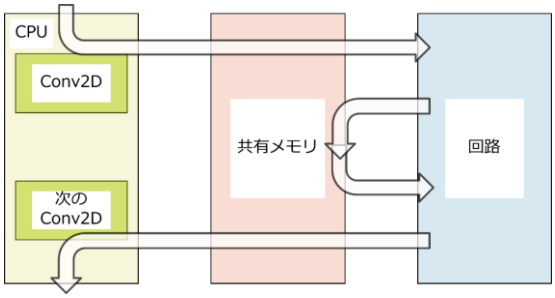

図 3.2　手法適用後のデータの流れ

また，ResNet50 の構造上，ある層の処理を終えるとそれより前の層のデータが不必要となる場合がある．共有メモリのデータの内，必要なくなったデータは，該当するメモリ領域を解放し，その箇所に新たなデータを書き込むことで削除しており，共有メモリのリソースを無駄なく使用している．

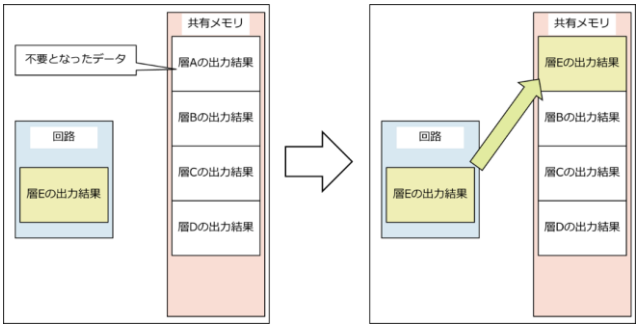

図 3.3　共有メモリの不要となったデータの削除

4. 検証方法

検証方法は，図 3.1 のデータ共有方式を用いている先行研究[1]の場合と，図 3.2 の提案手法を適用した場合の 2 つを Ultra96-V2 上で動作させ，それぞれの全体の推論時間，データ共有に要した時間，層間のデータ受け渡し時間を算出．提案手法の適用前と適用後の時間を比較することで，効果を確認する．

†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

5. 結果

表 5.1 は手法適用前と適用後の全体の推論処理時間と各データ転送に要した時間を示したものである．

表 5.1　提案手法の適用前と適用後の処理時間

	適用前	適用後	差分
全体の推論時間	4635[ms]	2732[ms]	1903[ms]
CPU→共有メモリ	586[ms]	212[ms]	374[ms]
共有メモリ→CPU	447[ms]	120[ms]	327[ms]
層→層	896[ms]	37[ms]	859[ms]

表 5.1 の結果から，適用前と比べて全体の推論処理時間が 1903ms，約 1.7 倍速くなっていることがわかる．

また，今回の手法を適用した後の全体の推論時間の内訳を図 5.1 に示す．

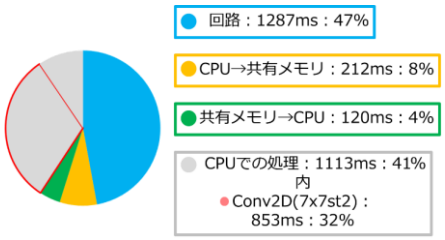

図 5.1　適用後の時間の内訳

6. おわりに

今回は，回路による処理が連続する場合におけるデータ転送の無駄を減らすことで，時間の短縮を図った．結果として，手法適用前と比べて約 1.7 倍速く処理を終えることができた．

今後の展望として，現在 CPU で処理しているカーネルサイズ 7x7，ストライド 2 の Conv2D 層を回路化することによる処理の高速化を考えている．なぜなら，図 5.1 より，この Conv2D 層は全体の 32%を占めているからである．また，これまでは Ultra96-V2[3]をエッジデバイスとして使用していたが，他のエッジデバイスを用いた場合の変化についても調査していくことを考えている．

参考文献

[1] 村井稔大，中西知嘉子，“エッジ AI のデータ分割による転送手法の改善とさらなる高速化手法の検討”，電子情報通信学会総合大会(2024).
[2] Deep Residual Learning for Image Recognition, <https://arxiv.org/abs/1512.03385>
[3] AVNET, ULTRA96-V2 <https://www.avnet.com/opasdata/d120001/medias/docus/193/5365-pb-ultra96-v2-v4a.pdf>
[4] Vitis HLS , <https://japan.xilinx.com/member/forms/download/xef.html?filename=Xilinx_Unified_2022.2_1014_8888_Lin64.bin>
[5] Vivado Design Suite , <https://japan.xilinx.com/member/forms/download/xef.html?filename=Xilinx_Unified_2022.2_1014_8888_Lin64.bin>
[6] 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"，電子情報通信学会ソサイエティ大会(2021)