C-008

画像認識モデル ResNet50 の推論をエッジデバイス上で高速に処理する手法の検討

Investigation of a method to process the inference of the ResNet50 image recognition model on edge devices at high speed.

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 を動作させる「エッジ AI」が注目されている．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．

そこで本研究では，深層学習モデルの 1 つである「ResNet」に注目し，エッジデバイスを用いた推論処理の高速化を目指す．デバイスとして SoC FPGA を採用した．また，ベース回路として先行研究[3]で作成された汎用的な Conv2D 回路を使用した．ResNet の構造の特徴を調べ，その特徴にあった専用の回路を実装することで推論処理の高速化を実現する．

高速化が実現出来ていることを確認するために，CPU のみを用いた場合の処理時間や先行研究[3]の回路を用いた場合の処理時間，本研究で高速化した処理時間を比較する．

2. 使用機器・ネットワークモデルの分析

本研究では，SoC FPGA ボードである Ultra96V2 をエッジデバイスとして使用し，画像認識によく用いられる深層学習モデル ResNet50 の推論処理の高速化を行う．

2.1 Ultra96V2

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[1]．SoC FPGA は，CPU と FPGA が同一チップ上に存在するデバイスである．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ことで高速化を図ることができる．

2.2 ResNet50

ResNet（Residual Network）とは，Microsoft 社の Kaiming He 氏が考案したニューラルネットワークモデルである[2]．ResNet にはパラメータ数が異なるモデルが複数あるが，今回はエッジデバイスへの搭載を視野に入れているため，精度やモデルサイズのバランスがよい ResNet50 を採用した．

ResNet50 を Ultra96V2 の CPU のみで動かし，各層の時間を計測，回路化する箇所を分析した．表 2.2.1 に ResNet50 を CPU のみで動かした際の各層の時間と割合を示す．

表 2.2.1　CPU のみでの各層の処理時間

層の種類	処理時間（ms）	割合（%）
Conv2D	145343.997	98.26
Batch Normalization	844.526	0.57
Add	546.314	0.37
Activation	452.462	0.31
Others	728.3572	0.49

表 2.2.1 から，Conv2D 層が約 98%と最も大きな時間を要していることがわかった．そこで，ResNet50 の層の内，Conv2D 層とその周りの構造に注目し，特徴を調べた．ResNet50 の Conv2D 層の次には必ず Batch Normalization 層がある．さらにその後ろに Activation 層である ReLU が存在する場合がある．このことから，Conv2D 層と Batch Noramalization 層，Activation 層を 1 つに結合して回路化し，FPGA に処理させることで高速化を行うことにした．ただし，Activation 層は存在しない場合があるため，選択的に実行できるように実装した．

3. 回路作成

3.1 ベース回路

本研究では，回路化するためのベース回路として，先行研究[3]で作成された汎用的な Conv2D 回路を使用した．この回路は Conv2D 層の処理を FPGA にさせることで高速化に成功したものである．この回路が対応している Conv2D 層の畳み込み処理の種類を表 3.1.1 に示す．

表 3.1.1　先行研究[3]の回路が対応している Conv2D 処理

カーネルサイズ	ストライド 1	ストライド 2
1×1	○	○
3×3	○	○
7×7	×	×

ResNet50 にはカーネルサイズ 7×7，ストライド 2 の Conv2D 層が 1 つだけあるが，先行研究の回路には対応していないため，この層に関しては CPU で処理する．

3.2 回路設計

ベース回路の Conv2D 層の処理に加え，Batch Normalization 層と ReLU 層の処理を結合し，追加実装した．回路設計を行うツールとして Vitis HLS 2020.2，RTL を回路に組み込むツールとして Vivado 2020.2 を用いた．

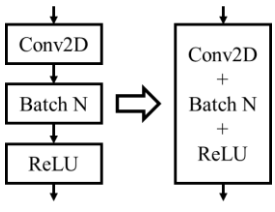

図 3.2.1　回路化した箇所のイメージ図

図 3.2.1 は本研究で回路化した箇所のイメージである．先行研究[3]では Conv2D 層のみを回路に実装していたが，Batch Normalization 層や ReLU 層の処理を一つにまとめることで層数を減らし，回路で行うことで高速化する．また，FPGA に供給するクロック周波数は 300MHz とした．

3.3 データ転送

転送するデータは float 型であるため，CPU と回路間のデータパス幅は 64bit とし，1 度に 32bit のデータを 2 つ送信するようにした．また，CPU と FPGA 間のデータのやりとりを行うために共有メモリを使用した．共有メモリを使用するためのデバイスドライバとしては，User space mappable DMA Buffer[4]を採用し，共有メモリから FPGA へのデータ転送方式には DMA 転送を採用した．DMA 転送とは，FPGA とメモリ間のデータ転送を，CPU を介さずに行う方式である．先行研究[3]では，推論時に推論に必要なデータを共有メモリに書き込んでいた．しかし，この方法ではデータの書き込みに時間を要してしまう．そこで，重みなどの変化しないデータはモデル読み込み時に予め共有メモリに書き込み，DMA 転送の開始アドレスとサイズを CPU で制御することで，推論の処理時間を減らした．

3.4 データ分割

回路には保持できるデータ量に限界があり，Conv2D 層のサイズによっては，その限界を超えることがある．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た．表 3.4.1 は回路のデータサイズ上限一覧である．

表 3.4.1　回路のデータサイズ上限一覧

カーネルサイズ	データの種類	データサイズ	
		先行研究[3]	本研究
カーネル 1×1	入力データ	66	56
	チャネル	576	576
	カーネル	152	128
カーネル 3×3	入力データ	66	56
	チャネル	64	64
	カーネル	152	128

回路のデータサイズ上限は先行研究[3]の上限を元にしている．しかし，先行研究[3]の上限のままでは ResNet50 の入力データの上限より大きい部分があるため，その箇所を変更した．具体的には，ResNet50 の内，カーネルサイズが 3×3，1×1 の場合の入力データサイズは 56×56 が最大であるため，上限を 56 に変更した．また，ResNet50 のカーネル数は最低が 64 で 2 の倍数であるため，128 とした．データサイズの上限を超える層を処理する場合は，CPU 上でデータを分割して回路にデータを送信し，順次演算を行った後，演算結果を CPU 上で合成する．これによって回路の上限より大きなサイズの層を処理する場合でも回路を使用することができる．ただし，Batch Normalization 層の処理は Conv2D 層のチャネル方向の積和に対して行われるため，チャネルの分割が発生すると Batch Normalization 層の処理を行えない．よって，チャネルの分割が発生した場合は Batch Normalization 層以降の処理は CPU で行う．

4. 検証方法

検証方法として ResNet50 を CPU のみで推論させた場合と，先行研究[3]で作成された回路を用いて推論させた場合，本研究で作成した回路を用いて推論させた場合の 3 つの処理時間を測定し，比較することで高速化ができているかを確認した．測定範囲は推論時間全体とする．また，本研究で作成した回路を用いて推論させた場合については，回路実行の時間，CPU から共有メモリへの書き込み時間，共有メモリから CPU への読み込み時間を含めた処理時間の内訳も測定した．測定はそれぞれ 5 回行い，平均の時間を比較した．

5. 結果

CPU のみで推論させた場合，先行研究で作成された回路を用いて推論させた場合，本研究で作成した回路を用いて推論させた場合の 3 つの結果を表 4.2.1 に示す．

表 4.2.1　推論全体の処理時間

	CPU のみ	先行研究	本研究
全体の処理時間	147914.7[ms]	8218.7[ms]	5446.9[ms]

表 4.2.1 より，本研究で作成した回路は，CPU のみの場合に比べて約 27.1 倍，先行研究と比べて約 1.5 倍の高速化に成功したことがわかる．本研究で作成した回路を用いた場合の時間の内訳を図 4.2.1 に示す．

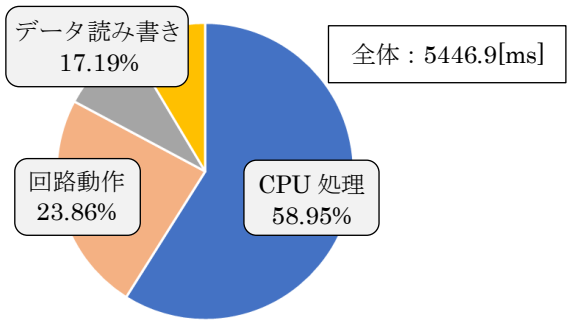

図 4.2.1　作成した回路を用いた場合の時間の内訳

6. 結論

本研究では，SoC FPGA を用いて，ResNet50 の高速化を行った．ResNet50 の構造の特徴から，連続して処理が行われる Conv2D 層，Batch Normalization 層，Activation 層を 1 つの層と見なし，回路に実装した．結果，CPU のみの実行に比べ，約 27.1 倍の高速化に成功した．

今後の展望として，他の層の回路化による高速化を実現していくことを考えている．なぜなら，図 4.2.1 より，CPU の処理時間が約 60%を占めており，この部分が最も大きな高速化を見込めるからである．特に今回回路に実装しなかったカーネルサイズ 7×7 の Conv2D 層は約 854[ms]と全体の約 15.6%を占めている．また，Add 層も約 550[ms]と大きな時間を要しているので，これらの回路化による高速化などを検討し，実装していく．

参考文献

[1] AVNET, ULTRA96-V2,
<https://www.avnet.com/opasdata/d120001/medias/docus/193/5365-pb-ultra96-v2-v4a.pdf>
[2] Deep Residual Learning for Image Recognition
<https://arxiv.org/abs/1512.03385>
[3] 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).
[4] ikwzm, udmabuf,
<https://github.com/ikwzm/udmabuf>

† 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