C-014

FPGA を用いた CNN の構成要素のハードウェア化
Hardware implementation of CNN components using FPGA

西森 祐介† 森下 賢幸† 小椋 清孝† 伊藤 信之†
Yusuke Nishimori Takayuki Morishita Kiyotaka Komoku Nobuyuki Itoh

1. はじめに

CNN(畳み込みニューラルネットワーク)は画像認識において高い性能を示すが、大規模なデータを扱うと規模が大きくなり学習や認識に時間がかかる。本研究では、CNN を FPGA 上でハードウェアに実装することで高速化を図る。ハードウェア化することにより、パイプライン処理や並列処理が可能となり、処理時間が短縮され高速化できると期待される。本研究では、CNN を構成する要素である畳み込み演算処理回路とプーリング演算処理回路及び認識回路・学習回路を設計し、回路構成および速度・規模の評価を行った。開発環境は Xilinx 社の ISE Design Suite 14.7 を使用し、設計言語には Verilog HDL を用いた。

2. 畳み込み演算処理回路

画像処理などによく用いられる畳み込みニューラルネットは、畳み込み層とプーリング層と全結合層で構成される[1]。畳み込み演算では、入力された画像の特徴を抽出する。畳み込みは画像とフィルタを重ね合わせ、重なり合う画素どうしの積を求め、フィルタ全体でその積の合計値を求め出力する仕組みである。図 1 に畳み込み演算の概要を示す。入力データを 11×11、フィルタを 3×3 とする。

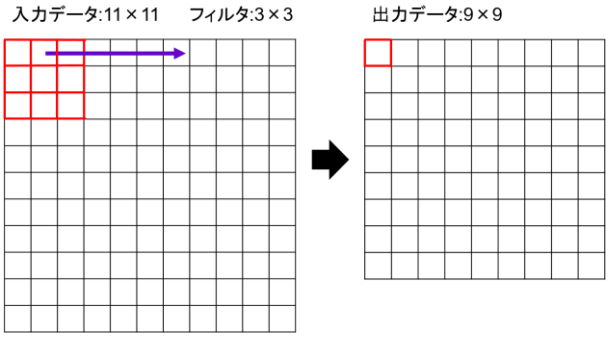

図 1　畳み込み処理

この畳み込み演算処理を回路にする。畳み込み演算処理回路の動作の概要を図 2 に示す。

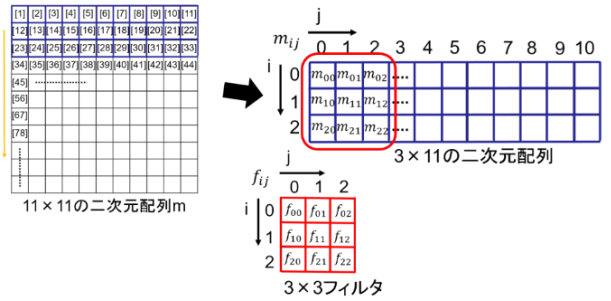

図 2　畳み込み演算処理回路の仕組み

入力画像は 11×11 で、縦方向成分を i、方向成分を j とした 11×11 の二次元配列m_{ij}を用意する。フィルタは 3×3 を用いるため同じ方向成分の 3×11 の二次元配列を用意する。3×11 の二次元配列に 11×11 の青色で示された入力画像データを挿入して畳み込み処理のフィルタリングを行う。図 2 の畳み込み演算の出力をy_{00}とすると計算結果は$y_{00} = m_{00} \times f_{00} + m_{01} \times f_{01} + m_{02} \times f_{02} + m_{10} \times f_{10} + m_{11} \times f_{11} + m_{12} \times f_{12} + m_{20} \times f_{20} + m_{21} \times f_{21} + m_{22} \times f_{22}$となる。3×11 の二次元配列でこの計算処理を横方向に 1 ずつずらしながら行う。3×11 の二次元配列に渡す入力画像データを 11×11 の二次元配列で縦方向に下に 1 ずつずらしながら、11×11 の入力画像データ全てにおいてこれと同様の畳み込み処理を行う。

3. プーリング演算処理回路

プーリング処理に用いる手法として最大プーリングを用いる。プーリング処理を行うことで局所的なデータの不変性を得ることができる。図 3 に最大プーリングの概要を示す。

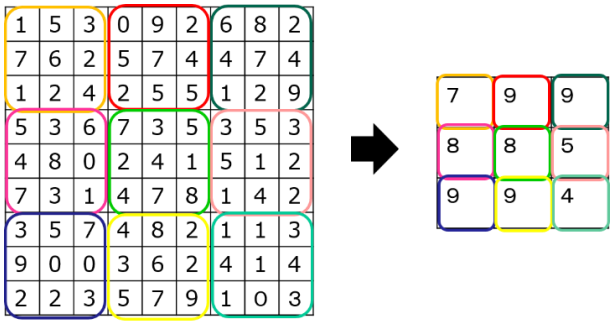

図 3　最大プーリング処理

図 3 に示すように、最大プーリングとは入力画像にフィルタをかけ、その中の最大値を出力する。出力画像サイズはフィルタのサイズを H×H とすると、H^2となる。図 4 に設計した最大プーリング処理回路の概要を示す。

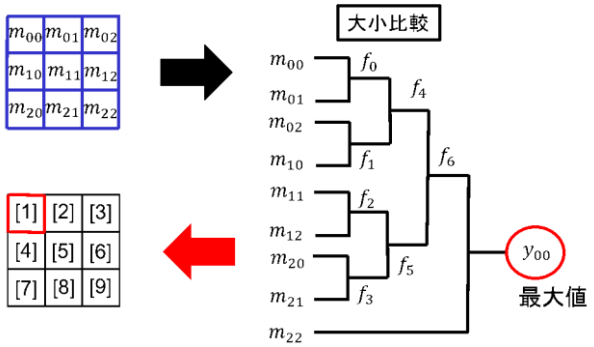

図 4　最大プーリング演算処理回路の仕組み

畳み込み処理と同様、プーリング本体の上位モジュールと演算を行う下位モジュールに分けて考える。上位モジュールに関しては畳み込み処理と同様の記述方法を用いる。畳み込み処理の出力画像データサイズは 9×9 であるため、

畳み込み後に行われるプーリング処理の入力画像サイズは 9×9 となる。フィルタに関しては同じく 3×3 のものを用いる。最大プーリングとはフィルタ内の最大値を出力する仕組みであるため、図 4 に示すように 3×3 の 9 個のデータを二値ずつ順に大小比較を施していく。これにより求められたy_{00}がこのフィルタ内の最大値となる。

4. 全結合層の処理回路

全結合層はシナプスとシグモイド関数で構成されたニューロンを回路化した。入力層で畳み込みとプーリングによる前処理を行い、そこからニューロンで構成された中間層と出力層の順にデータを出力する。中間層の数は 3 つ、出力層の数は 1 つとする。学習は誤差逆伝搬法を用いて行い、これらの回路を設計した。図 5 に設計した出力層の重み学習回路のブロック図を示す。

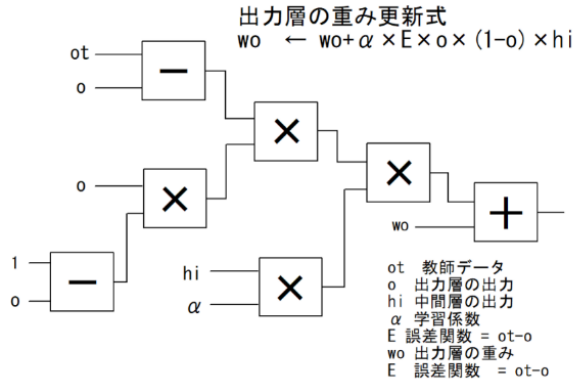

図 5　出力層の重み学習回路ブロック図

図 5 に示すように誤差逆伝搬法の計算式を回路化することにより学習計算を行う。これについてもシミュレーションによる動作検証を行う。

5. シミュレーション結果

設計した畳み込み演算処理回路とプーリング演算処理回路の動作検証を行った。図 6 に畳み込み演算処理回路を示し、図 7 にプーリング演算処理回路を示す。入力データは 11×11 のデータの中央の列を 1 行から 11 行まですべて 1 にしたものをシミュレーションの入力値とし、3×3 のフィルタに関しても中央の列を全て 1 にしたものをシミュレーションに用いた。

図 6　畳み込み層出力結果

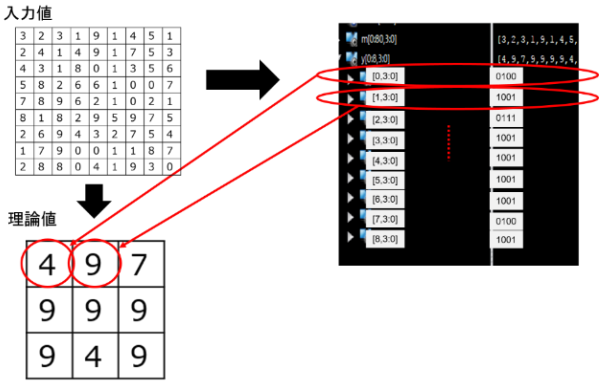

図 7　プーリング層出力結果

図 6 の出力データを見ると中央の列が全て 3 になっている。これは理論値と一致している。図 6 の赤枠で示した[4,3:0]を例にとると、これは左側の数値の 4 が 9×9 の出力データの配列内の番号を示し、右側の 3:0 は 4bit であることを示している。$0011_{(2)} = 3_{(10)}$であるためこれは理論値と一致している。図 7 の最大プーリングでは正しく動作されているかわかりやすくするために入力データをランダムな入力値とした。得られたシミュレーション結果の出力値を 10 進数に変換すると理論値と一致していることが確認できた。

次に、畳み込み層とプーリング層の回路規模の評価を行った。ターゲットデバイスを Xilinx xc5vlx220-2ff1760 とした時の畳み込み層の回路規模は 4536LUTs、プーリング層の回路規模は 333LUTs であった。さらに畳み込み層での値が入力されてから出力されるまで計算の遅延時間は 127.0ns、プーリング層では 17.6ns となった。シミュレーションには Isim の Post-Route Simulation を用いた。

6. おわりに

畳み込み層、プーリング層、全結合層及び出力層の重み学習回路を回路設計し、正しく動作していることを確認した。CNN の中間層の重み学習回路が設計完了後に全体の速度評価を行う予定である。今後の課題は、回路規模の評価や大規模 CNN へ対応することである。

参考文献

[1] 小高知宏, “機械学習と深層学習”, オーム社 (2016).

†岡山県立大学大学院　情報系工学研究科　システム工学専攻　GRADUATE SCHOOL OF COMPUTER SCIENCE AND SYSTEMS ENGINEERING,OKAYAMA PREFECTUAL UNIVERSITY