Tạp chí Phát triển Khoa học và Công nghệ – Kĩ thuật và Công nghệ, 5(1):1332-1341

Open Access Full Text Article

Bài nghiên cứu

Tối ưu các mạng nơ-ron tích chập trên phần cứng có tài nguyên giới hạn

Phạm Văn Khoa[1,*], Trần Nhật Quang[2], Nguyễn Ngô Lâm[3]

Use your smartphone to scan this QR code and download this article

TÓM TẮT

Các mạng nơ-ron tích chập (CNN) đóng vai trò rất quan trọng trong nhiều ứng dụng thị giác máy tính như phân loại, nhận dạng đối tượng. Để đạt được hiệu quả nhận dạng cao, thông thường các mạng nơ-ron này cần được thực thi trên các nền tảng tính toán có hiệu năng cao với tốc độ xử lý nhanh và không gian bộ nhớ lớn. Điều này là một trở ngại rất lớn đối với ứng dụng chạy trên các thiết bị tính toán có tài nguyên phần cứng bị giới hạn như các máy tính nhúng. Ở các lớp tích chập, để có thể trích xuất được đặc trưng của đối tượng ngõ vào cần thiết phải thực thi một lượng lớn các phép nhân và cộng dồn. Bên cạnh đó, hoạt động nhân trên các số có dấu chấm động yêu cầu thời gian tính toán lớn cũng như phần cứng phức tạp. Nghiên cứu này phân tích và chỉ rõ những nguyên nhân làm giới hạn hiệu năng tính toán của mạng CNN. Từ đó, trình bày phương pháp để thực thi các mạng tích chập trên phần cứng có tài nguyên giới hạn. Việc đánh giá hiệu năng về mặt công suất, thời gian thực thi cũng như tỉ lệ nhận dạng được trình bày chi tiết thông qua mô phỏng và thực thi trên phần cứng. Các kết quả thực nghiệm trên cả hai nền tảng FPGA và bộ xử lý nhúng ARM Cortex-A chỉ ra rằng mạng CNN sử dụng phương pháp XNOR-popcount có thể được tối ưu để đạt hiệu năng tính toán tăng 1000 lần và công suất tiêu thụ giảm xấp xỉ 24 lần khi so sánh với mạng CNN thông thường trên các bộ xử lý nhúng.

Từ khoá: mạng nơ-ron tích chập, phép nhân, hoạt động nhân chập, XNOR-popcount, CIFAR-10, ảnh trên giây, PYNQ-Z2

[1]*Khoa Đào tạo Quốc tế, Trường Đại học Sư Phạm Kỹ Thuật, Tp Hồ Chí Minh, Việt Nam*

[2]*Khoa Công nghệ Thông tin, Trường Đại học Sư Phạm Kỹ Thuật, Tp Hồ Chí Minh, Việt Nam*

[3]*Khoa Đào tạo Chất lượng cao, Trường Đại học Sư Phạm Kỹ Thuật, Tp Hồ Chí Minh, Việt Nam*

Liên hệ

Phạm Văn Khoa, Khoa Đào tạo Quốc tế, Trường Đại học Sư Phạm Kỹ Thuật, Tp Hồ Chí Minh, Việt Nam

Email: khoapv@hcmute.edu.vn

Lịch sử

- Ngày nhận: 23-8-2021
- Ngày chấp nhận: 22-02-2022
- Ngày đăng: 31-3-2022

DOI : 10.32508/stdjet.v4i4.906

Check for updates

GIỚI THIỆU

Mạng nơ-ron tích chập (Convolutional neural network – CNN) là một mô hình mạng nơ-ron nhân tạo có độ chính xác cao, được triển khai nhiều trong các ứng dụng nhận dạng ảnh và giọng nói[1–3]. Để đạt được độ chính xác theo yêu cầu đặt ra, mô hình mạng CNN thông thường phải thực hiện một lượng lớn các phép toán với kiểu dữ liệu dấu chấm động (floating-point numbers)[1–3]. Chính vì điều đó, mạng CNN thường yêu cầu phần cứng phức tạp với tốc độ xử lý nhanh và không gian bộ nhớ lớn để huấn luyện và thực thi tác vụ nhận dạng. Được minh họa trong Hình 1a, mạng CNN bao gồm nhiều lớp được tổ chức liên tiếp nhau gồm lớp tích chập (convolutional layer), lớp tổng hợp (pooling layer) và lớp kết nối đầy đủ (fully-connected layer). Khối lượng tính toán của mạng CNN chủ yếu nằm ở các phép toán tích chập (convolution), gồm phép nhân và phép cộng dồn, để trích đặc trưng các đối tượng ảnh ngõ vào.

Trong tác vụ nhận dạng ảnh được thực thi bởi các mạng CNN, phép tích chập giữa một phần của ảnh ngõ vào (receptive field) và bộ lọc (kernel) có thể thực hiện bằng phép nhân ma trận trên từng phần tử (element–wise product)[1]. Để đạt được tỉ lệ nhận dạng ảnh cao, thông thường mạng CNN được tạo bởi nhiều lớp tích chập ghép liền nhau trong đó các trọng số huấn luyện được biểu diễn với dữ liệu dấu chấm động (floating-point number). Việc thực hiện phép toán tích chập với phép nhân và cộng dồn (Multiply–Accumulate - MAC), trên các dữ liệu dấu chấm động yêu cầu phải cần một phần cứng tính toán phức tạp và công suất tiêu thụ lớn.

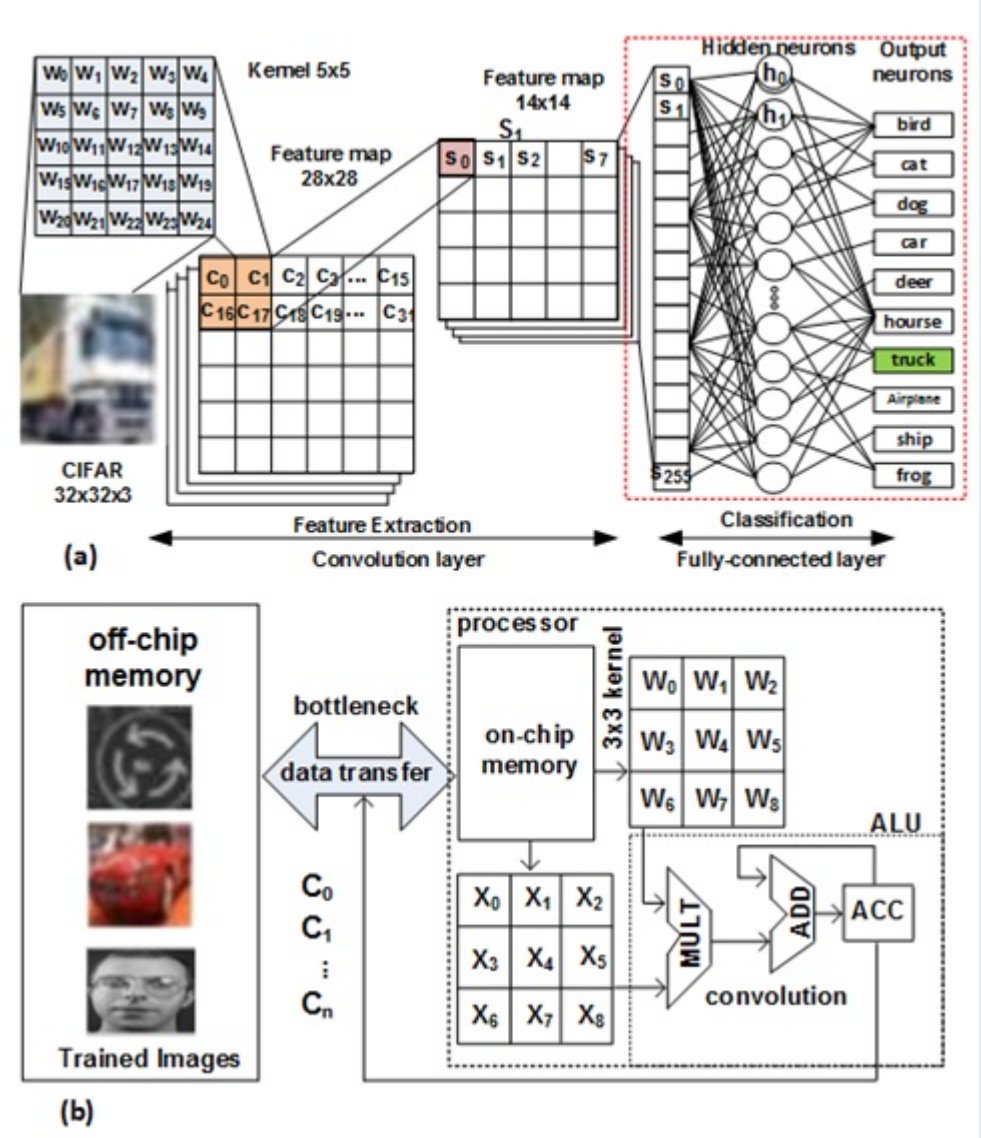
Để thực thi mạng CNN với phương pháp trên, thông thường các máy tính có hiệu năng cao được sử dụng để đạt được yêu cầu về tính toán độ chính xác cao lẫn tốc độ xử lý nhanh. Điều này là một trở ngại rất lớn đối với các thiết bị nhúng với cấu hình phần cứng thấp[4]. Các thiết bị này thường có một bộ xử lý và bộ nhớ giới hạn để thỏa mãn yêu cầu về chi phí sản xuất cũng như tối ưu năng lượng tiêu thụ. Ngày nay, nhiều phương pháp đã được đề xuất để có thể hiện thực hóa việc thực thi các mạng nơ-ron nhân tạo trên các phần cứng có tài nguyên giới hạn này[5–13].

MẠNG NƠ-RON TÍCH CHẬP

Phép toán nhân cộng dồn

Việc thực thi phép toán tích chập giữa ảnh ngõ vào và bộ lọc để trích được đặc trưng của đối tượng được thể

Trích dẫn bài báo này: Khoa P V, Quang T N, Lâm N N. **Tối ưu các mạng nơ-ron tích chập trên phần cứng có tài nguyên giới hạn**. *Sci. Tech. Dev. J. - Eng. Tech.;* 5(1):1332-1341.

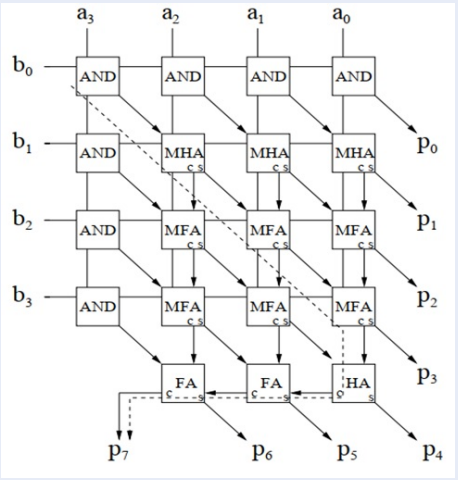


Hình 1: a) Kiến trúc mạng CNN điển hình; b) Hoạt động của phép tích chập giữa ảnh ngõ vào và bộ lọc kích thước 3 x 3

hiện trong Hình 1b. Bằng cách trượt bộ lọc qua ảnh ngõ vào và thực hiện các phép toán nhân và cộng dồn, lớp tích chập thu được kết quả là các bản đồ đặc trưng (feature map) của ảnh ngõ vào. Ứng với mỗi bộ lọc và đặc điểm của ảnh ngõ vào (hoặc feature map nếu là đầu ra của lớp tích chập trước đó) ta cần tính một bản đồ đặc trưng. Như vậy có thể thấy để thu được các bản đồ đặc trưng của một ảnh ngõ vào thì trên một lớp tích chập cần phải thực hiện rất nhiều phép toán tích chập sử dụng các phép nhân và cộng dồn [1,5]. Bên cạnh đó, đối với các tác vụ nhận dạng trong thực tế thì ảnh ngõ vào và các bản đồ đặc trưng thường có kích thước lớn. Vì thế, thông thường hệ thống yêu cầu một không gian lưu trữ lớn chứa tập dữ liệu các ảnh và các tham số mạng thu được trong và sau quá trình huấn luyện. Hiện nay, đa phần các máy tính được thiết kế theo kiến trúc Von-Neumann [14]. Hạn chế của kiến trúc này khi thực hiện phép toán tích chập trên dữ liệu có độ phức tạp cao là xảy ra hiện tượng thắt cổ chai vì sự tách biệt giữa bộ xử lý và bộ nhớ [14]. Như vậy, phép tính tích chập trên các kiến trúc Von-Neumann sẽ ảnh hưởng lớn đến thời gian xử lý hay tốc độ huấn luyện lẫn nhận dạng của mạng.

Giải pháp tăng tần số của bộ xử lý và mở rộng không gian vùng nhớ trên bộ xử lý hầu như không phù hợp với các nền tảng phần cứng cấu hình thấp. Nghiên

cứu[6–13] chỉ ra rằng việc tính toán trên số nguyên (integer) đạt được lợi điểm về mặt thời gian xử lý và hiệu quả năng lượng hơn nhiều lần so với các phép xử lý trên số thực (floating-point number). Bên cạnh đó, phép xử lý bit và phép cộng là các hoạt động tính toán yêu cầu phần cứng đơn giản và thời gian thực thi nhanh đáng kể so với phép nhân và chia.

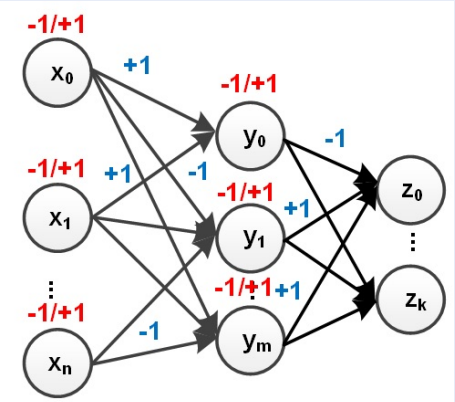


Hình 2: Cấu trúc mạch nhân 2 số nguyên 4 bit[15]

Trong hệ thống máy tính, phép nhân được phát triển từ thiết kế của mạch cộng bán phần và toàn phần, vì thế mạch nhân có cấu trúc phần cứng phức tạp và tiêu tốn thời gian xử lý lớn[15–17]. Hình 2 minh họa thiết kế điển hình cho mạch nhân 2 số 4 bit[15]. Có thể dễ nhận thấy rằng thiết kế mạch nhân khá phức tạp dựa trên nền tảng phép cộng và dịch dữ liệu. Vì thế, để có thể hiện thực hóa việc thực thi các mạng CNN trên các phần cứng có tài nguyên giới hạn như các thiết bị điện tử nhúng cần phải phân tích cụ thể hoạt động của phép tính tích chập và nghiên cứu, đề xuất các giải pháp tối ưu hóa thiết kế phần cứng thực thi phép tính này trên các mạng CNN.

Lượng tử hóa và phương pháp tính toán trên số nguyên

Các nghiên cứu trước đây đã khảo sát sự phân bố giá trị của các tham số huấn luyện và tín hiệu kích hoạt ngõ ra trong mạng nơ-ron nhân tạo[6–13]. Mục đích của các nghiên cứu này nhằm để giản lược số bit cần thiết để biểu diễn các giá trị này. Điều này giúp giảm bớt yêu cầu về không gian vùng nhớ chứa các tham số huấn luyện cũng như các phần cứng xử lý tính toán. Các nghiên cứu đã chỉ ra rằng tham số huấn luyện và cả tín hiệu kích hoạt đều có thể được lượng tử hóa với số bit nhỏ hơn nhiều so với việc sử dụng số có dấu chấm động kích thước lớn như 32 bit hoặc 64 bit. Kết quả là các hệ thống nhận dạng sử dụng các giá trị lượng tử hóa (Quantized Neural Networks – QNN) ra đời[6–9]. Các hệ thống này có ưu điểm lớn là tốc độ xử lý nhanh và thiết kế phần cứng giản lược hơn, nhưng nhược điểm là tỉ lệ nhận dạng thường thấp hơn các hệ thống sử dụng số dấu chấm động. Tuy nhiên, bằng cách sử dụng các giải thuật lượng tử hóa phù hợp, sự chênh lệch này ở mức chấp nhận được.



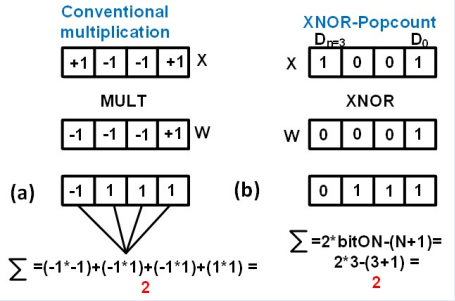
Hình 3: Mạng nơ-ron nhị phân

Mạng nơ-ron nhị phân (Binarized Neural Networks – BNN)[10–12] là một trường hợp đặc biệt của QNN khi cả tham số huấn luyện và tín hiệu kích hoạt đều được lượng tử thành các giá trị nhị phân như minh họa trong Hình 3. Như vậy, trong quá trình huấn luyện mạng, thuật toán sẽ thay đổi giá trị của các tham số để trở thành -1 hoặc +1. Hàm kích hoạt (activation function) trong mạng BNN sử dụng là Sign(x) thay vì sử dụng các hàm phức tạp khó được thực thi với phần cứng đơn giản như Sigmoid hay ReLU. Hàm Sign(x) được sử dụng để xét dấu cho kết quả phép nhân cộng dồn (x) sao cho thỏa công thức (1)

$$x^b = Sign = \begin{cases} +1, \ if \ x \geq 0 \\ -1, \ if \ x < 0 \end{cases} \quad (1)$$

Như vậy ngõ ra của hàm kích hoạt Sign(x) có giá trị -1 hoặc +1 như được minh họa trong Hình 3. Bằng việc lượng tử hóa này, phép tích chập gồm phép nhân và cộng dồn sẽ yêu cầu phần cứng đơn giản hơn so với việc tính toán trên giá trị số dấu chấm động và kích thước lớn. Ví dụ, với trường hợp giá trị ngõ vào **X[3:0]** = {1; -1; -1; 1} và tham số huấn luyện **W[3:0]** = {-1; -1; -1; 1} như thể hiện trong Hình 4a, ta cần sử dụng 4 phép nhân và phép cộng số 2 bit có dấu để thu được kết quả phép tích chập có giá trị là 2. Tuy nhiên, không gian vùng nhớ để chứa các tham số huấn luyện

và phần cứng để xử lý phép tích chập có thể được giản lược tối đa chỉ sử dụng 1 bit nếu thực hiện thao tác mã hóa nhằm loại bỏ bit dấu chuyển giá trị +1 và -1 thành 1 và 0 tương ứng như thể hiện trong Hình 4b.



Hình 4: Phương pháp thực thi và kết quả của nhân cộng dồn các số nhị phân 1, -1 sử dụng: a) phương pháp MAC; b) phương pháp XNOR-popcount

Ngoài các lệnh xử lý toán học như cộng, trừ, nhân, chia, thiết kế bên trong của khối tính toán số học trong các bộ xử lý máy tính cũng chứa các phép xử lý bit hay còn gọi là bitwise. Các phép tính trên bit này được cấu tạo từ phần cứng là các cổng luận lý (logic gates). Các cổng luận lý này được kết hợp để cấu thành nên các thiết kế tính toán phức tạp như các mạch tổ hợp, tuần tự và các bộ điều khiển. Vì thế, xét về hiệu năng công suất tiêu thụ và thời gian tính toán thì hoạt động xử lý bit sẽ mang lại hiệu năng tốt hơn[17]. Phương pháp xử lý XNOR-popcount[13] được trình bày trong nghiên cứu này nhằm thay thế hoàn toàn cho phép tích chập khi cả tham số huấn luyện và tín hiệu ngõ vào được mã hóa nhị phân thành 0 hoặc 1. Phương pháp xử lý XNOR-popcount được thể hiện theo công thức (2) dưới đây.

$$X.W = 2\sum_{i=1}^{N} XNOR(X_i, W_i) - N \quad (2)$$

Trong đó, **X** là vec-tơ của ảnh ngõ vào hoặc ngõ ra của hàm kích hoạt. **W** là vec-tơ được tạo thành từ các tham số huấn luyện. **N** là độ dài véc-tơ. Thiết kế kiến trúc XNOR-popcount với 3 thao tác chính gồm (1) thực hiện xử lý XNOR trên 2 số nhị phân 1 bit, (2) thực hiện Popcount đếm tổng số bit 1 có trong kết quả XNOR, và (3) thực hiện **2**× **S** – **N** trong đó **S** là tổng số bit 1, **N** là chiều dài cố định của vec-tơ. Như minh họa trong Hình 4b, phần cứng cần thiết để xử lý XNOR-popcount gồm các cổng XNOR 2 ngõ vào 4 bit, mạch cộng tính tổng các bit 1 của kết quả XNOR, mạch dịch trái dữ liệu nhằm thực hiện phép nhân 2, và mạch trừ 2 số 4 bit. Đặc biệt để giản lược tối đa về mặt phần cứng thì toán hạng 2 trong hoạt động trừ có thể được đặt cố định bởi giá trị **N** là kích thước của bộ lọc và đã được định sẵn. So sánh với phương pháp nhân cộng dồn MAC, phần cứng XNOR-popcount ở Hình 4b có phần cứng được giản lược nhưng kết quả tính toán tương đương.

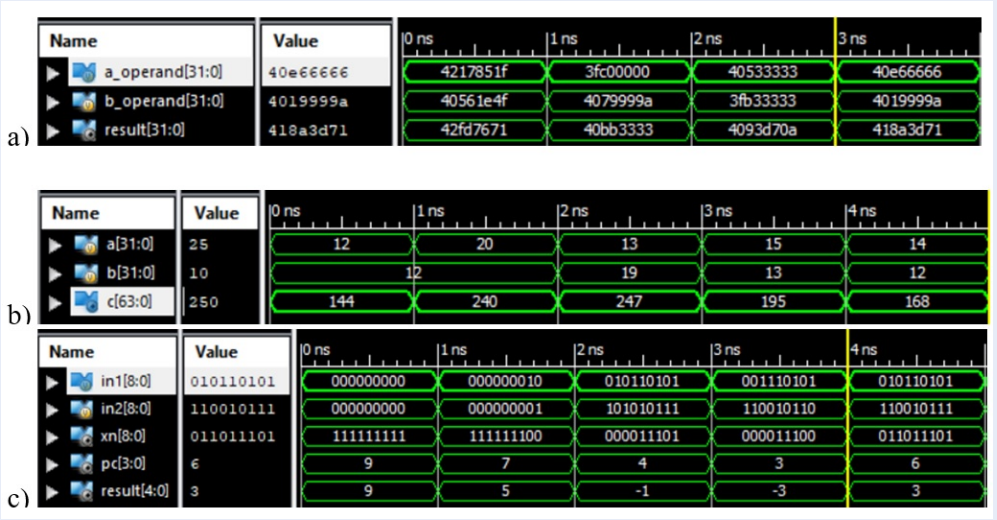
PHƯƠNG PHÁP

Thực nghiệm với phép nhân cộng dồn trên số có dấu chấm động (FP-32) và số nguyên (INT-32)

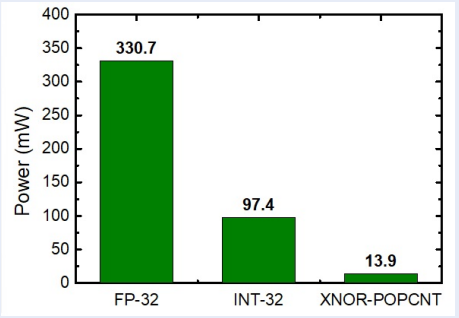
Để thực nghiệm sự ảnh hưởng của phép nhân số thực, số nguyên và phương pháp thay thế XNOR-popcount lên hiệu năng của hệ thống, nghiên cứu này sẽ thực thi và đánh giá các thiết kế mạch nhân với số thực, số nguyên và phương pháp thay thế sử dụng các hoạt động tính toán trên bit. Hình 5 thể hiện dạng sóng của các hoạt động tính toán nêu trên. Giả sử rằng trong mạng CNN, các phép tích chập được thực hiện trên ma trận 2 chiều của ảnh ngõ vào và bộ lọc với kích thước 3×3. Trong phương pháp truyền thống áp dụng trên các mạng CNN, cần thiết phải sử dụng 9 bộ nhân số 32 bit dấu chấm động hoặc số nguyên và các mạch cộng dồn kết quả. Tuy nhiên, với phương pháp XNOR-popcount chỉ cần sử dụng các mạch xử lý bit và dịch cũng như mạch trừ số 9 bit.

Dạng sóng của thiết kế XNOR-popcount hoạt động ở tần số 1GHz, được minh họa trong Hình 5c. Tại thời điểm t = 1ns, giá trị ngõ vào thứ nhất là **in1**[8:0] = "000000010" và ngõ vào thứ hai là **in2**[8:0] = "000000001", kết quả phép XNOR được thể hiện với thanh ghi 8 bit **xn**[8:0] = "111111100". Thanh ghi 4 bit **pc**[3:0] được sử dụng để lưu trữ số bit '1' có trong kết quả XNOR. Cuối cùng, thanh ghi 4 bit **result** [3:0] chứa kết quả XNOR-popcount. Trong trường hợp trên thanh ghi result có giá trị là '5' tương đương với kết quả có được từ phương pháp nhân cộng dồn khi hai ngõ vào lần lượt là **in1**[8:0] = {-1; -1; -1; -1; -1; -1; -1; -1; +1; -1} và **in2**[8:0] = {-1; -1; -1; -1; -1; -1; -1; -1; +1}.

Từ kết quả phân tích công suất được tổng hợp bằng phần mềm ISE Xilinx XPower[18], giá trị công suất tiêu thụ của từng thiết kế được thể hiện trong Hình 6. Công suất tiêu thụ tại tần số 100Mhz của hoạt động MAC trên 2 vec-tơ 9 bit với số có dấu chấm động (FP-32), với số nguyên (INT-32) và phương pháp thay thế MAC sử dụng XNOR-popcount được thể hiện trong Hình 6. Có thể thấy thiết kế MAC với số 32 bit dấu chấm động tiêu thụ công suất lớn nhất là 330.7mW. Khi thực hiện MAC trên số nguyên 32 bit, công suất giảm đi 3 lần còn 97.4mW. Đáng chú ý, công suất tiêu thụ có thể giảm gần 24 lần từ 330.7mW xuống 13.9mW khi sử dụng phương pháp XNOR-popcount với phần cứng đơn giản.



Hình 5: Dạng sóng của thiết kế mạch nhân 2 số với a) số dấu chấm động 32 bit b) số nguyên 32 bit c) XNOR-popcount trên 2 vec-tơ 9 bit



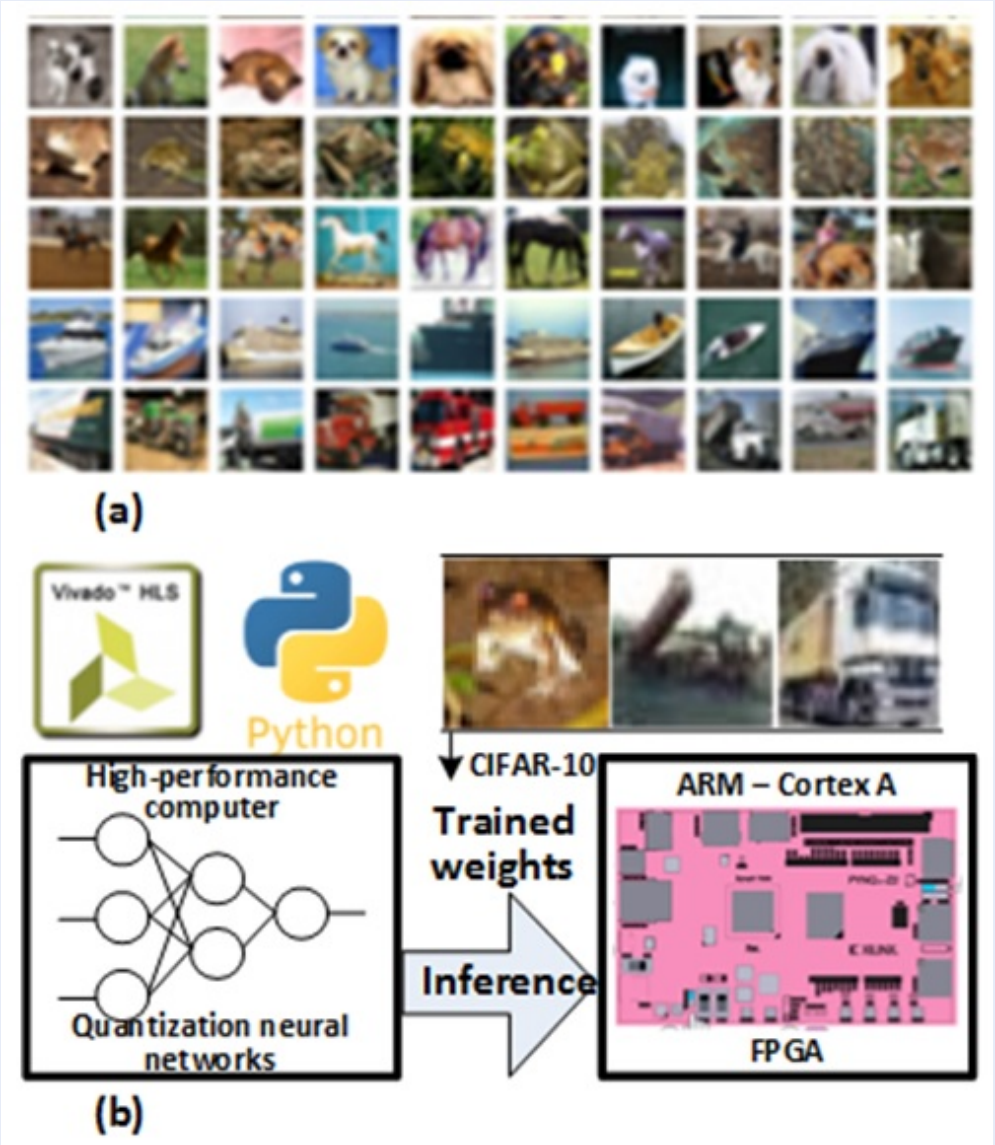
Hình 6: Công suất tiêu thụ của hoạt động nhân 2 ma trận kích thước 3×3 tại tần số f=100Mhz trên từng thiết kế phần cứng

Thực nghiệm trên mạng nơ-ron tích chập với tập dữ liệu CIFAR-10

Nhằm đánh giá hiệu năng về thời gian xử lý của phương pháp XNOR-popcount khi thay thế hoạt động MAC trên các mạng CNN thông thường, nghiên cứu này đã thực nghiệm một số mạng CNN với cấu hình phần cứng khác nhau để thực thi tác vụ nhận dạng đối tượng trên dữ liệu ảnh màu CIFAR-10 sử dụng nền tảng FPGA. Trong đó, CIFAR-10 là một tập dữ liệu ảnh màu phổ biến trong lĩnh vực thị giác máy tính[19]. Tập dữ liệu này chứa 60.000 ảnh với 50.000 ảnh làm tập dữ liệu huấn luyện và 10.000 ảnh làm tập dữ liệu kiểm tra. Được minh họa chi tiết trong Hình 7a, các ảnh có kích thước 32×32×3 pixels được phân bố đều cho 10 lớp tương ứng với 10 đối tượng nhận dạng như máy bay, xe hơi, tàu thủy, xe tải…

Hình 7b thể hiện sơ đồ khối hệ thống được sử dụng trong nghiên cứu này. Trong đó, mạng CNN sử dụng kiến trúc được mô tả như trong Bảng 1 để nhận diện các ảnh trong tập CIFAR-10. Trước tiên, nghiên cứu sẽ huấn luyện mạng CNN với các trọng số sử dụng giá trị số dấu chấm động, các trọng số và giá trị ngõ ra của hàm kích hoạt sau đó được lượng tử hóa với số bit cố định để tạo ra các cấu hình mạng CNN khác nhau. Toàn bộ quá trình huấn luyện và lượng tử hóa được khảo sát và thực hiện trên các máy tính hiệu năng cao với ngôn ngữ cấp cao Python. Với sự hỗ trợ của công cụ tổng hợp phần cứng Vivado của hãng Xilinx, các cấu hình mạng CNN khác nhau sử dụng phương pháp nhân cộng dồn (MAC) thông thường và phương pháp thay thế XNOR-popcount được cấu hình một cách linh hoạt lên phần cứng FPGA PYNQ-Z2[20].

Với nền tảng phần cứng PYNQ, nhà thiết kế có thể viết mã trực tiếp bằng ngôn ngữ cấp cao Python để thực thi mạng CNN trên cả bộ xử lý 32-bit ARM Cortex-A và phần cứng FPGA với cấu hình mô tả chi tiết trong Bảng 2. Sự kết hợp này mang lại một giải pháp hiệu quả để thực thi và đánh giá hiệu năng của các mạng CNN trên các cấu hình phần cứng khác nhau một cách linh hoạt mà không đòi hỏi nhà phát triển phải có kiến thức chuyên sâu về thiết kế phần cứng. Dựa trên đặc điểm của nền tảng PYNQ, các mạng CNN với trọng số và giá trị hàm kích hoạt được lượng tử được thực thi trên cả bộ xử lý ARM và phần cứng FPGA một cách nhanh chóng. Từ đó, việc đánh giá hiệu năng của thiết kế bao gồm tốc độ xử lý, tỉ lệ nhận dạng và tài nguyên phần cứng được phân tích



Hình 7: Đánh giá hiệu năng nhận dạng của mạng CNN với a) bộ ảnh CIFAR-10; b) phần cứng PYNQ-Z2

một cách cụ thể. Ngày nay, một số công cụ đã được giới thiệu để hỗ trợ việc chuyển đổi mã lập trình cấp cao như Python cho thực thi các mạng CNN trên bộ xử lý đa dụng sang mã mô tả phần cứng nhằm thực thi các mạng CNN trên phần cứng FPGA [21,22].

Kiến trúc của mạng CNN gồm 6 lớp tích chập (convolution) và 3 lớp kết nối đầy đủ (fully connected) với hơn 10 triệu tham số sử dụng trong nghiên cứu này được thể hiện chi tiết trong Bảng 1. Trong đó, các lớp tích chập sử dụng các bộ lọc có kích thước 3×3. Có thể thấy rằng, các lớp tích chập trong mạng có số lượng tham số ít hơn so với các lớp kết nối đầy đủ bởi các tham số này được chia sẻ trong mỗi lớp tích chập. Tuy nhiên, các lớp tích chập yêu cầu số lượng lớn các phép nhân ma trận giữa kernel và receptive field. Để đáp ứng được yêu cầu nhận dạng đối tượng thời gian thực thông thường phần cứng phép nhân và cộng dồn phải được thực thi với tần số cao. Điều này là một trở ngại rất lớn cho các thiết kế hệ thống nhúng khi bị giới hạn bởi tần số hoạt động và công suất tiêu thụ.

Nghiên cứu này đánh giá các thiết kế gồm mạng CNN sử dụng số dấu chấm động 32 bit (CNN-FP32), mạng CNN được lượng tử hóa ở cấp độ nhị phân với 1 bit cho tham số huấn luyện và tín hiệu kích hoạt (QNN-W1A1); 2 bit cho tham số huấn luyện và tín hiệu kích hoạt (QNN-W2A2); 1 bit cho tham số huấn luyện và

Bảng 1: Kiến trúc mạng cnn nhận dạng các đối tượng trong tập dữ liệu CIFAR-10

Lớp mạng	Bộ lọc	Số lượng tham số
Convolution	3×3	3×3×3×128 = 3,456
Convolution	3×3	3×3×128×128 = 147,456
Convolution	3×3	3×3×128×256 = 294,912
Convolution	3×3	3×3×256×256 = 589,824
Convolution	3×3	3×3×256×512 = 1,179,648
Convolution	3×3	3×3×512×512 = 2,359,296
Fully connected		1024×4608 = 4,718,592
Fully connected		1024×1024 = 1,048,576
Fully connected		1024×10 = 10,240
Tổng cộng		10,352,000

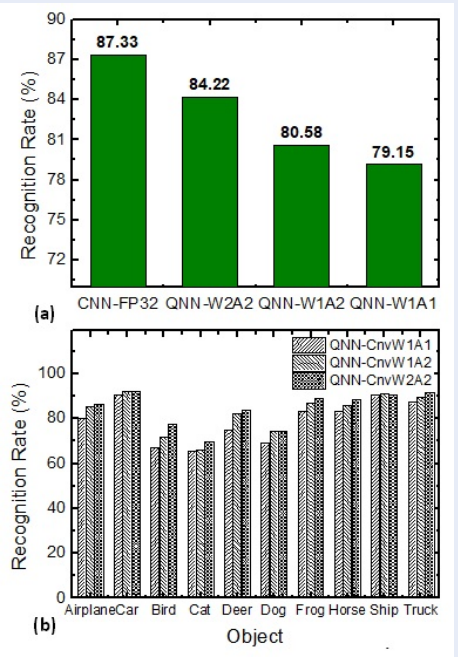
Bảng 2: Cấu hình phần cứng ARM và FPGA

	ARM	FPGA
Dòng sản phẩm	Cortex-A9	Artix-7
Tài nguyên xử lý	Lõi kép	215 ngàn cổng logic
Tần số hoạt động	650 MHz	100 Mhz
Bộ nhớ SRAM	8 MB	630 KB

2 bit cho tín hiệu kích hoạt (QNN-W1A2). Thiết kế của mạng QNN-W1A1 còn có tên khác là mạng nơ-ron nhị phân (Binarized Neural Network - BNN), đây là trường hợp đặc biệt của mạng QNN khi cả tham số huấn luyện và tín hiệu kích hoạt đều được thể hiện bằng giá trị nhị phân.

KẾT QUẢ

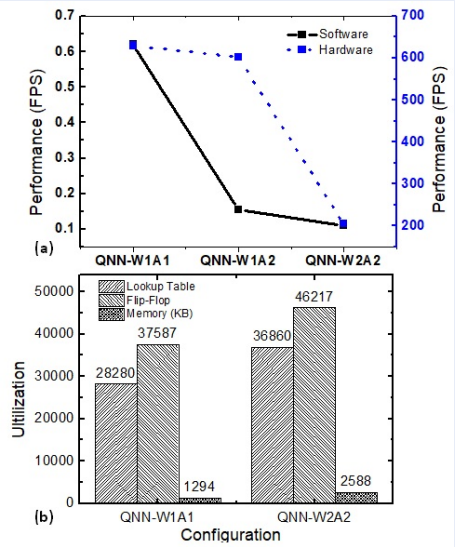
Hình 8a minh họa tỉ lệ nhận dạng của các thiết kế mạng CNN. Trong đó mạng CNN-FP32 sử dụng hoạt động MAC thông thường với số dấu chấm động đạt được tỉ lệ nhận dạng cao nhất với 87.32%. Mạng QNN-W1A1 với các tham số huấn luyện được lượng tử hóa với 1 bit có tỉ lệ nhận dạng giảm khoảng 8% còn 79.15%. Khi thực hiện nâng số bit lượng tử hóa cho cả tham số huấn luyện và tín hiệu kích hoạt lên thành 2 bit với QNN-W2A2, tỉ lệ nhận dạng tăng lên gần 5% từ 79.15% lên 84.22% nhưng độ phức tạp về phần cứng xử lý và không gian bộ nhớ lưu trữ cũng tăng theo. Đặc biệt, đối với cấu hình QNN-W1A2, tỉ lệ nhận dạng là 80.58% xấp xỉ với cấu hình QNN-W1A1. Điều này cho thấy số bit biểu diễn cho ngõ ra nơ-ron hay tín hiệu kích hoạt trên các mạng CNN không ảnh hưởng lớn đến tỉ lệ nhận dạng so với các trọng số huấn luyện. Kết quả nhận dạng trên các đối tượng của CIFAR-10 với từng thiết kế mạng QNN được minh họa chi tiết trong Hình 8b.



Hình 8: a) Tỉ lệ nhận dạng của mạng CNN sử dụng cấu hình phần cứng khác nhau; b) Tỉ lệ nhận dạng của 10 đối tượng trong tập dữ liệu CIFAR-10 trên 3 cấu hình CNN

Phương pháp XNOR-popcount với phần cứng đơn giản giúp tăng tốc độ thực thi của lớp tích chập trong mạng CNN. Nghiên cứu này cũng đã so sánh hiệu năng thực thi các cấu hình mạng khác nhau trên cả bộ xử lý nhúng và phần cứng tái cấu hình trên FPGA được tích hợp trên nền tảng PYNQ. Hình 9a minh họa hiệu năng xử lý của các phương pháp thực thi mạng CNN trên các nền tảng phần cứng khác nhau khi xét thời gian cần thiết để nhận dạng từng ảnh ngõ vào. Trong đó đường đứt nét thể hiện cho hiệu năng xử lý của phần cứng FPGA và đường liền nét thể hiện cho hiệu năng các mạng CNN thực thi trên bộ xử lý ARM-Cortex A. Hiệu năng xử lý được đo lường bằng thông số khung hình trên giây (frame per second - FPS).

Từ kết quả Hình 9a, có thể thấy rằng sự đánh đổi về tỉ lệ nhận dạng đã mang lại cải thiện rất lớn về tốc độ xử lý. Thiết kế phần cứng XNOR-popcount khi thực thi mạng QNN-W1A1 với tham số huấn luyện và tín hiệu kích hoạt là 1 bit nhị phân đạt hiệu năng cao



Hình 9: a) Hiệu năng xử lý đo lường bằng FPS đối với mạng CNN trên các nền tảng và cấu hình phần cứng khác nhau b) Tài nguyên phần cứng được sử dụng trên FPGA gồm Lookup Table, Flip-flop và bộ nhớ cần thiết để thực thi mạng QNN-W1A1 và QNN-W2A2

nhất khi có thể nhận dạng được 630 khung hình/giây. Trong khi đó, mạng QNN-W2A2 với tham số huấn luyện và tín hiệu kích hoạt là 2 bit nhị phân cần sử dụng phần cứng MAC để xử lý số nguyên 2 bit có hiệu năng giảm hơn 3 lần khi chỉ nhận dạng được 205 khung hình/giây bởi mạng này yêu cầu sử dụng các phép nhân số nguyên có độ phức tạp cao hơn so với phương pháp XNOR-popcount. Trong trường hợp thực thi mạng CNN trên bộ xử lý ARM-Cortex A, hiệu năng xử lý ghi nhận chỉ đạt 0.6 khung hình/giây đối với cấu hình QNN-W1A1 và khi sử dụng cấu hình QNN-W2A2 hiệu năng giảm đi rất nhiều chỉ còn 0.1 khung hình/giây.

Dựa trên Bảng I, ta có số lượng tham số của kiến trúc mạng CNN là 10,352,000. Nếu các tham số này được biểu diễn bởi các giá trị dấu chấm động 32 bit thì bộ nhớ tối thiểu được yêu cầu tương ứng là: 10,352,000 $\times$ 32 = 331,264,000 bit = 41,408 KByte. Tuy nhiên, mạng QNN-W1A1 với các trọng số và giá trị hàm kích hoạt được lượng tử 1 bit kích thước lưu trữ tối thiểu là: 10,352,000 bit = 1,294 KB tương ứng giảm 32 lần so với mạng CNN không lượng tử. Thêm vào đó, kiến trúc mạng CNN được sử dụng trong nghiên cứu này yêu cầu một lượng tài nguyên rất lớn trên FPGA để thực thi và không phù hợp với cấu hình phần cứng FPGA được hỗ trợ trên PYNQ. Kết quả tại Hình 8a cho thấy khi so sánh với QNN-W2A2 thì BNN hay QNN-W1A1 có tỉ lệ nhận dạng giảm khoảng 5% từ 84.22% xuống 79.15% nhưng đem lại tốc độ xử lý nhận dạng nhanh hơn gấp 3 lần tương ứng là 630 khung hình/giây so với chỉ 205 khung hình/giây đạt được bởi QNN-W2A2 như thể hiện trong Hình 9a. Hiệu năng xử lý tăng bắt nguồn từ thiết kế phần cứng của mạng QNN-W1A1 đơn giản hơn so với QNN-W2A2. Hình 9b thể hiện tài nguyên phần cứng trên FPGA cần thiết kế xây dựng 2 mạng trên bao gồm các thành phần logic của mạch tổ hợp, tuần tự cũng như bộ nhớ lưu trữ.

THẢO LUẬN

Mục đích của nghiên cứu này nhằm chỉ ra những điểm hạn chế liên quan đến thời gian thực thi và công suất tiêu thụ khi máy tính sử dụng phương pháp nhân và cộng dồn trên các dữ liệu phức tạp có dấu chấm động để xử lý tính toán các tác vụ nhận dạng với mạng nơ-ron nhân tạo. Ở nền tảng kết nối vạn vật (Internet of Thing – IoT) các thiết kế máy tính đa phần được nhúng vào trong các hệ thống thông minh và được đặt phân tán ở nhiều nơi khác nhau sử dụng nguồn năng lượng giới hạn từ pin hoặc các nguồn năng lượng tái tạo nên công suất tiêu thụ là một vấn đề rất quan trọng cần phải xem xét. Bên cạnh đó, các máy tính nhúng với tài nguyên phần cứng giới hạn đa phần sẽ hoạt động với chức năng phân loại dữ liệu thô cảm nhận từ môi trường bên ngoài thông qua các cảm biến trước khi đưa dữ liệu đã được xử lý cơ bản lên các máy tính có hiệu năng cao hơn như các máy chủ. Vì thế, thời gian thực thi cần đáp ứng nhanh để cập nhật kịp thời với sự thay đổi liên tục từ môi trường bên ngoài.

Việc so sánh và phân tích hiệu quả tính toán trên hai nền tảng bộ xử lý nhúng ARM và phần cứng thay thế được thực thi trên FPGA là nhằm chỉ ra rõ sự không phù hợp dẫn đến một hiệu năng thực thi thấp trên các máy tính nhúng ARM khi thực thi các cấu trúc mạng nơ-ron kích thước lớn với dữ liệu huấn luyện phức tạp. Giải pháp đưa ra là kết hợp song song phương pháp lượng tử hóa dữ liệu huấn luyện và phương pháp xử lý bit XNOR popcount thay thế cho phương pháp nhân cộng dồn thông thường thì hiệu năng tính toán của hệ thống được cải thiện đáng kể với tỉ lệ nhận dạng chấp nhận được trên các máy tính nhúng.

Nghiên cứu này đã chỉ ra nếu áp dụng giải pháp nêu trên thì thời gian thực thi có thể giảm 1000 lần ở tỉ lệ nhận dạng xấp xỉ 80% khi so sánh với phương pháp sử dụng nhân cộng dồn với dữ liệu có dấu chấm động trên thiết kế máy tính nhúng ARM-Cortex A. Bên cạnh đó, một kết quả thực nghiệm trước đó từ hãng công nghệ ARM đã đưa ra đánh giá hiệu năng của mạng CNN trên thiết kế máy nhúng ARM-Cortex M. Nghiên cứu của Liangzhen Lai và các cộng sự[23] đã

sử dụng tập dữ liệu CIFAR-10 và cho khả năng nhận dạng được 10 khung hình/giây tại tần số hoạt động 216Mhz khi các trọng số huấn luyện được lượng tử hóa thành các giá trị 8-bit.

KẾT LUẬN

Nghiên cứu này phân tích và chỉ rõ những nguyên nhân làm giới hạn hiệu năng của mạng nơ-ron tích chập. Phép nhân tích chập vốn yêu cầu thời gian tính toán đáng kể, là nguyên nhân chính dẫn đến kém hiệu quả cả về năng lượng tiêu thụ và thời gian thực thi tác vụ nhận dạng của CNN. Bằng việc phân tích các yếu tố về mặt công suất tiêu thụ và tốc độ xử lý trên mạng CNN với các thiết kế phần cứng khác nhau, có thể thấy việc giản lược hóa phần cứng để xử lý các dữ liệu số nguyên có kích thước nhỏ mang lại hiệu quả cao về hiệu năng. Trong đó, phương pháp XNOR-popcount với thiết kế phần cứng giản lược làm tăng tốc độ xử lý lên hơn 1000 lần, công suất tiêu thụ giảm 24 lần với tỉ lệ nhận dạng chấp nhận được khi so với mạng CNN sử dụng hoạt động MAC thông thường trên các bộ xử lý nhúng thông thường.

LỜI CÁM ƠN

Nghiên cứu này thuộc đề tài năm 2022 được hỗ trợ kinh phí bởi Trường Đại học Sư Phạm Kỹ Thuật Tp. Hồ Chí Minh.

XUNG ĐỘT LỢI ÍCH

Nhóm tác giả xin cam đoan rằng không có bất kỳ xung đột lợi ích nào trong công bố bài báo.

ĐÓNG GÓP CỦA TÁC GIẢ

Phạm Văn Khoa là tác giả chính đưa ra ý tưởng và chủ đề nghiên cứu, mô phỏng, thực nghiệm các nội dung và viết bản thảo.

Trần Nhật Quang đã thảo luận về các vấn đề nghiên cứu và cùng tham gia viết bản thảo.

Nguyễn Ngô Lâm đã tham gia góp ý hiệu chỉnh bản thảo.

TÀI LIỆU THAM KHẢO

1. Lecun Y, et al. Gradientbased learning applied to document recognition. Proceedings of the IEEE. 1998;86:2278 – 2324. Available from: https://doi.org/10.1109/5.726791.
2. LeCun Y, et al. Deep learning. Nature. 2015;521(7553):436–444. PMID: 26017442. Available from: https://doi.org/10.1038/nature14539.
3. He K, et al. Deep Residual Learning for Image Recognition. arXiv:1512.03385, 2015;PMID: 26180094. Available from: https://doi.org/10.1109/CVPR.2016.90.
4. Chen J, Ran X. Deep Learning with Edge Computing: A Review. Proceedings of the IEEE. 2019;107(8). Available from: https://doi.org/10.1109/JPROC.2019.2921977.
5. Nurvitadhi E, et al. Can FPGAs Beat GPUs in Accelerating Next-Generation Deep Neural Networks? Proceedings of the 2017 ACM/SIGDA International Symposium on Field-Programmable Gate Arrays, California, USA, 2017;Available from: https://doi.org/10.1145/3020078.3021740.
6. Hubara I, et al. Quantized Neural Networks: Training Neural Networks with Low Precision Weights and Activations. arXiv:1609.07061, 2016;.
7. Han S, et al. Learning both Weights and Connections for Efficient Neural Network. arXiv:1506.02626, 2015;.
8. Wu J, et al. Quantized Convolutional Neural Networks for Mobile Devices. IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016;Available from: https://doi.org/10.1109/CVPR.2016.521.
9. Hwang K, Sung W. Fixed-point feedforward deep neural network design using weights +1, 0, and −1. Signal Processing Systems (SiPS), IEEE Workshop, Belfast, UK, 2014;Available from: https://doi.org/10.1109/SiPS.2014.6986082.
10. Qin H, et al. Binary Neural Networks: A Survey. arXiv:2004.03333, April 8, 2020;.
11. Simons T, Lee DJ. A Review of Binarized Neural Networks. Electrical and Computer Engineering, Brigham Young University, 12 June 2019;Available from: https://doi.org/10.3390/electronics8060661.
12. Courbariaux M, Bengio Y. Binarized Neural Networks: Training Neural Networks with Weights and Activations Constrained to +1 or −1. arXiv:1602.02830, 2016;.
13. Rastegariy M, et al. XNOR-Net: ImageNet Classification Using Binary Convolutional Neural Networks. Computer Vision - ECCV, pp. 525-542, 2016;Available from: https://doi.org/10.1007/978-3-319-46493-0_32.
14. Horowitz M. Computing's Energy Problem (and what we can do about it). presented at the Solid-State Circuits Conference Digest of Technical Papers (ISSCC), San Francisco, CA, USA, 2014;Available from: https://doi.org/10.1109/ISSCC.2014.6757323.
15. Mishra RS. Comparative Analysis of different Algorithm for Design of High-Speed Multiplier Accumulator Unit (MAC). Indian Journal of Science and Technology, vol. 9, iss. 8, pp. 1-5, 2016;Available from: https://doi.org/10.17485/ijst/2016/v9i8/83614.
16. Sumit M, et al. Design of efficient 64-bit MAC unit using vedic multiplier for DSP application - A review. Electronics and Communication Engineering, 2015;.
17. Stine JE. Digital Computer Arithmetic Datapath Design Using Verilog HDL. Springer. 2003;Available from: https://doi.org/10.1007/978-1-4419-8931-4.
18. Xilinx Power Tools Tutorial. 2010;Available from: https://www.xilinx.com/sw_manuals/xilinx14_7.
19. Krizhevsky A, Nair V, Hinton G. 2014;Available from: http://www.cs.toronto.edu/kriz/cifar.html.
20. Xilinx. PYNQ-Z2 Reference Manual v1.0. 2018;.
21. Zhang X, et al. AccDNN: An IP-Based DNN Generator for FPGAs. Annual International Symposium on Field-Programmable Custom Computing Machines (FCCM), USA, 2018;Available from: https://doi.org/10.1109/FCCM.2018.00044.
22. Xu P, et al. AutoDNNchip: An Automated DNN Chip Predictor and Builder for Both FPGAs and ASICs. ACM/SIGDA International Symposium on Field-Programmable Gate Arrays, USA, 2020;.
23. Lai L, et al. CMSIS-NN: Efficient Neural Network Kernels for Arm Cortex-M CPUs. arXiv:1801.06601. 2018;.

Science & Technology Development Journal – Engineering and Technology, 5(1):1332-1341

Open Access Full Text Article

Research Article

Optimizing the convolutional neural networks for resource-constraint hardwares

Van-Khoa Pham[1,*], Quang N. Tran[2], Ngo-Lam Nguyen[3]

Use your smartphone to scan this QR code and download this article

ABSTRACT

Convolutional neural networks (CNNs) play an important role in many computer vision applications such as object classification and recognition. To achieve high recognition rate, these neural networks are usually implemented on high-performance computing platforms with high processing speed and large memory. This is a big obstacle for deploying these models on devices with limited hardware resources such as embedded computers. For convolution layers, it requires a lot of multiply-accumulation operations to extract useful features from input images. Furthermore, multiplication of floating-point numbers has long latency and demands a big hardware overhead. In this paper, we analyze and identify the causes that limit the performance of CNNs. Then a method for implementing convolutional networks on hardware with limited resources is presented. Performance evaluation in terms of power, execution time as well as recognition rate is presented in detail. Experimental results on both the FPGA hardware platform and the ARM Cortex-A embedded processor indicate that CNNs using the XNOR-popcount approach can be optimized to achieve a 1000-fold increase in computational performance and approximately a 24-fold reduction in power consumption compared to the tranditional implementation of CNNs on common embedded computer systems.

Key words: convolution neural network, multiplication, convolution operation, XNOR-popcount, CIFAR-10, frame per second, PYNQ-Z2

[1]*Faculty of International Education, HCMC University of Technology and Education, Vietnam*

[2]*Faculty of Information Technology, HCMC University of Technology and Education, Vietnam*

[3]*Faculty for High Quality Training, HCMC University of Technology and Education, Vietnam*

Correspondence

Van-Khoa Pham, Faculty of International Education, HCMC University of Technology and Education, Vietnam

Email: khoapv@hcmute.edu.vn

History

- Received: 23-8-2021
- Accepted: 22-02-2022
- Published: 31-03-2022

DOI : 10.32508/stdjet.v4i4.906

Check for updates

Cite this article : Van-Khoa Pham, Nhat-Quang Nguyen, Ngo-Lam Nguyen. **Optimizing the convolutional neural networks for resource-constraint hardwares**. *Sci. Tech. Dev. J. – Engineering and Technology;* 5(1):1332-1341.