(19) 대한민국특허청(KR)
(12) 공개특허공보(A)

(11) 공개번호 10-2021-0014902
(43) 공개일자 2021년02월10일

(51) 국제특허분류(Int. Cl.)
G06N 3/063 (2006.01)
(52) CPC특허분류
G06N 3/063 (2013.01)
(21) 출원번호 10-2019-0092946
(22) 출원일자 2019년07월31일
심사청구일자 없음

(71) 출원인
삼성전자주식회사
경기도 수원시 영통구 삼성로 129 (매탄동)
(72) 발명자
김경훈
경기도 수원시 영통구 삼성로 129(매탄동)
김경훈
경기도 수원시 영통구 삼성로 129(매탄동)
(뒷면에 계속)
(74) 대리인
정홍식, 김태헌

전체 청구항 수 : 총 20 항

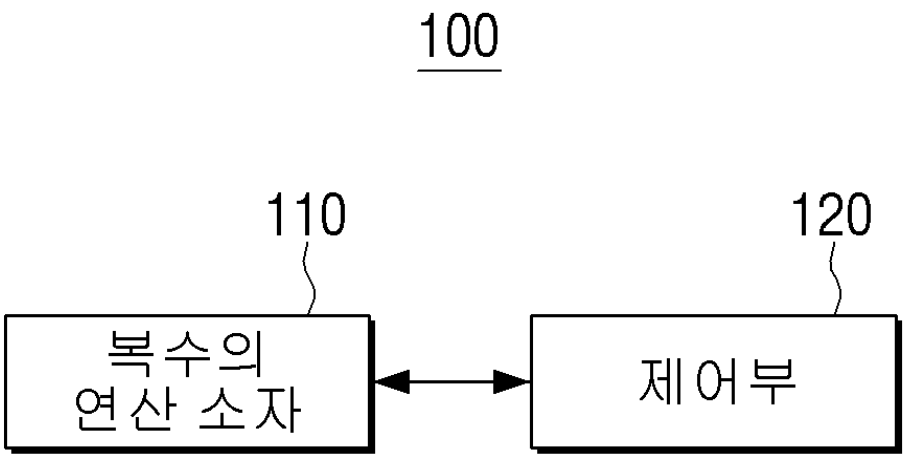
(54) 발명의 명칭 **프로세서 및 그 제어 방법**

(57) 요 약

프로세서가 개시된다. 본 프로세서는 매트릭스 형태로 배열된 복수의 연산 소자(Processing Element) 및 복수의 싸이클 동안 복수의 연산 소자를 제어하여 대상 데이터를 처리하는 제어부(controller)를 포함하며, 제어부는 대상 데이터에 포함된 복수의 엘리먼트 중 제1 로우(row)에 포함된 제1 엘리먼트들을 각각 복수의 연산 소자 중

(뒷면에 계속)

대 표 도 - 도2a



제1 로우에 배열된 제1 연산 소자들로 입력하며, 제1 연산 소자들 각각이, 인접한 제1 연산 소자로부터 제공된 데이터를 입력된 제1 엘리먼트와 연산하도록 제1 연산 소자들을 제어하고, 복수의 엘리먼트 중 제2 로우에 포함된 제2 엘리먼트들을 각각 복수의 연산 소자 중 제2 로우에 배열된 제2 연산 소자들로 입력하며, 제2 연산 소자들 각각이, 인접한 제2 연산 소자로부터 제공된 데이터를 입력된 제2 엘리먼트와 연산하도록 제2 연산 소자들을 제어하고, 제2 연산 소자들 각각이, 제1 연산 소자들 중 동일한 컬럼(column)에서 인접한 제1 연산 소자로부터 제공된 연산 데이터를 기저장된 연산 데이터와 연산하도록 제2 연산 소자들을 제어하여 대상 데이터에 대한 풀링(Pooling) 결과를 획득할 수 있다.

(72) 발명자

김현식
경기도 수원시 영통구 삼성로 129(매탄동)

송학섭
경기도 수원시 영통구 삼성로 129(매탄동)

위구연
경기도 수원시 영통구 삼성로 129(매탄동)

이정훈
경기도 수원시 영통구 삼성로 129(매탄동)

정진세
경기도 수원시 영통구 삼성로 129(매탄동)

조정욱
경기도 수원시 영통구 삼성로 129(매탄동)

한상복
경기도 수원시 영통구 삼성로 129(매탄동)

명 세 서

청구범위

청구항 1

매트릭스 형태로 배열된 복수의 연산 소자(Processing Element);

복수의 싸이클 동안 상기 복수의 연산 소자를 제어하여 대상 데이터를 처리하는 제어부(controller);를 포함하며,

상기 제어부는,

상기 대상 데이터에 포함된 복수의 엘리먼트 중 제1 로우(row)에 포함된 제1 엘리먼트들을 각각 상기 복수의 연산 소자 중 제1 로우에 배열된 제1 연산 소자들로 입력하며,

상기 제1 연산 소자들 각각이, 인접한 제1 연산 소자로부터 제공된 데이터를 상기 입력된 제1 엘리먼트와 연산하도록 상기 제1 연산 소자들을 제어하고, 상기 복수의 엘리먼트 중 제2 로우에 포함된 제2 엘리먼트들을 각각 상기 복수의 연산 소자 중 제2 로우에 배열된 제2 연산 소자들로 입력하며,

상기 제2 연산 소자들 각각이, 인접한 제2 연산 소자로부터 제공된 데이터를 상기 입력된 제2 엘리먼트와 연산하도록 상기 제2 연산 소자들을 제어하고,

상기 제2 연산 소자들 각각이, 상기 제1 연산 소자들 중 동일한 컬럼(column)에서 인접한 제1 연산 소자로부터 제공된 연산 데이터를 기저장된 연산 데이터와 연산하도록 상기 제2 연산 소자들을 제어하여 상기 대상 데이터에 대한 풀링(Pooling) 결과를 획득하는, 프로세서.

청구항 2

제1항에 있어서,

상기 제어부는,

제1 싸이클에서, 상기 제1 엘리먼트들을 각각 상기 제1 연산 소자들로 입력하며,

상기 제1 싸이클 직후의 제2 싸이클에서, 상기 제1 연산 소자들 각각이, 상기 인접한 제1 연산 소자로부터 제공된 데이터를 상기 입력된 제1 엘리먼트와 연산하도록 상기 제1 연산 소자들을 제어하고, 상기 제2 엘리먼트들을 각각 상기 제2 연산 소자들로 입력하며,

상기 제2 싸이클 직후의 제3 싸이클에서, 상기 제2 연산 소자들 각각이, 상기 인접한 제2 연산 소자로부터 제공된 데이터를 상기 입력된 제2 엘리먼트와 연산하도록 상기 제2 연산 소자들을 제어하고,

상기 제3 싸이클 직후의 제4 싸이클에서, 상기 제2 연산 소자들 각각이, 상기 제1 연산 소자들 중 상기 동일한 컬럼에서 인접한 제1 연산 소자로부터 제공된 연산 데이터를 상기 기저장된 연산 데이터와 연산하도록 상기 제2 연산 소자들을 제어하여 상기 풀링 결과를 획득하는, 프로세서.

청구항 3

제1항에 있어서,

상기 제어부는,

제1 싸이클에서, 상기 제1 엘리먼트들을 각각 상기 제1 연산 소자들로 입력하며,

상기 제1 싸이클 직후의 복수의 싸이클 동안, 상기 제1 연산 소자들 각각이, 상기 인접한 제1 연산 소자로부터 제공된 데이터를 상기 입력된 제1 엘리먼트와 연산하도록 상기 제1 연산 소자들을 제어하고,

상기 복수의 싸이클 중 하나인 제2 싸이클에서, 상기 제2 엘리먼트들을 각각 상기 제2 연산 소자들로 입력하며,

상기 제2 싸이클 직후의 복수의 싸이클 동안, 상기 제2 연산 소자들 각각이, 상기 인접한 제2 연산 소자로부터 제공된 데이터를 상기 입력된 제2 엘리먼트와 연산하도록 상기 제2 연산 소자들을 제어하고,

상기 제2 싸이클 직후의 복수의 싸이클로부터 기설정된 간격의 제3 싸이클에서, 상기 제2 연산 소자들 각각이, 상기 제1 연산 소자들 중 상기 동일한 컬럼에서 인접한 제1 연산 소자로부터 제공된 연산 데이터를 상기 기저장된 연산 데이터와 연산하도록 상기 제2 연산 소자들을 제어하여 상기 풀링 결과를 획득하며,

상기 제1 싸이클 직후의 복수의 싸이클의 개수와 상기 제2 싸이클 직후의 복수의 싸이클의 개수는 동일한, 프로세서.

청구항 4

제3항에 있어서,

상기 제어부는,

상기 제1 싸이클 직후의 복수의 싸이클 중 최초 싸이클에서, 상기 제1 연산 소자들 각각에 인접한 제1 연산 소자가 제1 엘리먼트를 제공하도록 상기 제1 연산 소자들을 제어하며,

상기 제2 싸이클 직후의 복수의 싸이클 중 최초 싸이클에서, 상기 제2 연산 소자들 각각에 인접한 제2 연산 소자가 제2 엘리먼트를 제공하도록 상기 제2 연산 소자들을 제어하는, 프로세서.

청구항 5

제4항에 있어서,

상기 제어부는,

상기 제1 싸이클 직후의 복수의 싸이클 중 최초 싸이클 이후의 싸이클 동안, 상기 제1 연산 소자들 각각에 인접한 제1 연산 소자가 직전 싸이클에서의 연산 데이터를 제공하도록 상기 제1 연산 소자들을 제어하며,

상기 제2 싸이클 직후의 복수의 싸이클 중 최초 싸이클 이후의 싸이클 동안, 상기 제2 연산 소자들 각각에 인접한 제2 연산 소자가 직전 싸이클에서의 연산 데이터를 제공하도록 상기 제2 연산 소자들을 제어하는, 프로세서.

청구항 6

제1항에 있어서,

상기 제어부는,

상기 대상 데이터에 포함된 복수의 엘리먼트 중 제3 로우에 포함된 제3 엘리먼트들을 각각 상기 복수의 연산 소자 중 제3 로우에 배열된 제3 연산 소자들로 입력하며,

상기 제3 연산 소자들 각각이, 인접한 제3 연산 소자로부터 제공된 데이터를 상기 입력된 제3 엘리먼트와 연산하도록 상기 제3 연산 소자들을 제어하고,

상기 제3 연산 소자들 각각이, 상기 제2 연산 소자들 중 동일한 컬럼에서 인접한 제2 연산 소자로부터 제공된 연산 데이터를 기저장된 연산 데이터와 연산하도록 상기 제3 연산 소자들을 제어하여 상기 풀링 결과를 획득하며,

상기 인접한 제2 연산 소자로부터 제공된 연산 데이터는,

상기 인접한 제2 연산 소자와 동일한 컬럼에서 인접한 제1 연산 소자로부터 제공된 상기 연산 데이터와 상기 인접한 제2 연산 소자의 상기 기저장된 연산 데이터의 연산 결과인, 프로세서.

청구항 7

제1항에 있어서,

상기 제어부는,

상기 제1 연산 소자들 각각이, 상기 인접한 제1 연산 소자로부터 제공된 데이터를 상기 입력된 제1 엘리먼트와 덧셈 연산하도록 상기 제1 연산 소자들을 제어하고,

상기 제2 연산 소자들 각각이, 상기 인접한 제2 연산 소자로부터 제공된 데이터를 상기 입력된 제2 엘리먼트와 덧셈 연산하도록 상기 제2 연산 소자들을 제어하며,

상기 제2 연산 소자들 각각이, 상기 제1 연산 소자들 중 상기 동일한 컬럼에서 인접한 제1 연산 소자로부터 제공된 연산 데이터를 상기 기저장된 연산 데이터와 덧셈 연산하도록 상기 제2 연산 소자들을 제어하여 평균값(average) 풀링 결과를 획득하는, 프로세서.

청구항 8

제1항에 있어서,

상기 제어부는,

상기 제1 연산 소자들 각각이, 상기 인접한 제1 연산 소자로부터 제공된 제1 엘리먼트를 상기 입력된 제1 엘리먼트와 비교 연산하도록 상기 제1 연산 소자들을 제어하고,

상기 제2 연산 소자들 각각이, 상기 인접한 제2 연산 소자로부터 제공된 제2 엘리먼트를 상기 입력된 제2 엘리먼트와 비교 연산하도록 상기 제2 연산 소자들을 제어하며,

상기 제2 연산 소자들 각각이, 상기 제1 연산 소자들 중 상기 동일한 컬럼에서 인접한 제1 연산 소자로부터 제공된 제1 엘리먼트를 기저장된 제2 엘리먼트와 비교 연산하도록 상기 제2 연산 소자들을 제어하여 최댓값(max) 풀링 결과를 획득하는, 프로세서.

청구항 9

제1항에 있어서,

상기 복수의 연산 소자 각각은,

제1 레지스터; 및

제2 레지스터;를 포함하고,

상기 제어부는,

상기 제1 연산 소자들 각각이, 상기 인접한 제1 연산 소자의 제1 레지스터 또는 제2 레지스터로부터 제공된 데이터를 상기 제1 연산 소자들 각각의 제1 레지스터에 저장된 제1 엘리먼트와 연산하여 상기 제1 연산 소자들 각각의 제2 레지스터에 저장하도록 상기 제1 연산 소자들을 제어하고,

상기 제2 연산 소자들 각각이, 상기 인접한 제2 연산 소자의 제1 레지스터 또는 제2 레지스터로부터 제공된 데이터를 상기 제2 연산 소자들 각각의 제1 레지스터에 저장된 제2 엘리먼트와 연산하여 상기 제2 연산 소자들 각각의 제2 레지스터에 저장하도록 상기 제2 연산 소자들을 제어하는, 프로세서.

청구항 10

제9항에 있어서,

상기 제어부는,

상기 제2 연산 소자들 각각이, 상기 제1 연산 소자들 중 상기 동일한 컬럼에서 인접한 제1 연산 소자의 제1 레지스터로부터 제공된 연산 데이터를 상기 제2 연산 소자들 각각의 제1 레지스터에 저장된 상기 연산 데이터와 연산하여 상기 제2 연산 소자들 각각의 제2 레지스터에 저장하도록 상기 제2 연산 소자들을 제어하는, 프로세서.

청구항 11

매트릭스 형태로 배열된 복수의 연산 소자(Processing Element)를 포함하며, 복수의 싸이클 동안 상기 복수의 연산 소자를 이용하여 대상 데이터에 대한 풀링(pooling)을 수행하는 프로세서의 제어 방법에 있어서,

상기 대상 데이터에 포함된 복수의 엘리먼트 중 제1 로우(row)에 포함된 제1 엘리먼트들을 각각 상기 복수의 연산 소자 중 제1 로우에 배열된 제1 연산 소자들로 입력하는 단계;

상기 제1 연산 소자들 각각이, 인접한 제1 연산 소자로부터 제공된 데이터를 상기 입력된 제1 엘리먼트와 연산하도록 상기 제1 연산 소자들을 제어하고, 상기 복수의 엘리먼트 중 제2 로우에 포함된 제2 엘리먼트들을 각각 상기 복수의 연산 소자 중 제2 로우에 배열된 제2 연산 소자들로 입력하는 단계;

상기 제2 연산 소자들 각각이, 인접한 제2 연산 소자로부터 제공된 데이터를 상기 입력된 제2 엘리먼트와 연산하도록 상기 제2 연산 소자들을 제어하는 단계; 및

상기 제2 연산 소자들 각각이, 상기 제1 연산 소자들 중 동일한 컬럼(column)에서 인접한 제1 연산 소자로부터 제공된 연산 데이터를 기저장된 연산 데이터와 연산하도록 상기 제2 연산 소자들을 제어하여 상기 풀링을 수행하는 단계;를 포함하는 제어 방법.

청구항 12

제11항에 있어서,

상기 제1 연산 소자들로 입력하는 단계는,

제1 싸이클에서, 상기 제1 엘리먼트들을 각각 상기 제1 연산 소자들로 입력하며,

상기 제2 연산 소자들로 입력하는 단계는,

상기 제1 싸이클 직후의 제2 싸이클에서, 상기 제1 연산 소자들 각각이, 상기 인접한 제1 연산 소자로부터 제공된 데이터를 상기 입력된 제1 엘리먼트와 연산하도록 상기 제1 연산 소자들을 제어하고, 상기 제2 엘리먼트들을 각각 상기 제2 연산 소자들로 입력하며,

상기 제2 연산 소자들을 제어하는 단계는,

상기 제2 싸이클 직후의 제3 싸이클에서, 상기 제2 연산 소자들 각각이, 상기 인접한 제2 연산 소자로부터 제공된 데이터를 상기 입력된 제2 엘리먼트와 연산하도록 상기 제2 연산 소자들을 제어하고,

상기 풀링을 수행하는 단계는,

상기 제3 싸이클 직후의 제4 싸이클에서, 상기 제2 연산 소자들 각각이, 상기 제1 연산 소자들 중 상기 동일한 컬럼에서 인접한 제1 연산 소자로부터 제공된 연산 데이터를 상기 기저장된 연산 데이터와 연산하도록 상기 제2 연산 소자들을 제어하여 상기 풀링을 수행하는, 제어 방법.

청구항 13

제11항에 있어서,

상기 제1 연산 소자들로 입력하는 단계는,

제1 싸이클에서, 상기 제1 엘리먼트들을 각각 상기 제1 연산 소자들로 입력하며,

상기 제2 연산 소자들로 입력하는 단계는,

상기 제1 싸이클 직후의 복수의 싸이클 동안, 상기 제1 연산 소자들 각각이, 상기 인접한 제1 연산 소자로부터 제공된 데이터를 상기 입력된 제1 엘리먼트와 연산하도록 상기 제1 연산 소자들을 제어하고,

상기 복수의 싸이클 중 하나인 제2 싸이클에서, 상기 제2 엘리먼트들을 각각 상기 제2 연산 소자들로 입력하며,

상기 제2 연산 소자들을 제어하는 단계는,

상기 제2 싸이클 직후의 복수의 싸이클 동안, 상기 제2 연산 소자들 각각이, 상기 인접한 제2 연산 소자로부터 제공된 데이터를 상기 입력된 제2 엘리먼트와 연산하도록 상기 제2 연산 소자들을 제어하고,

상기 풀링을 수행하는 단계는,

상기 제2 싸이클 직후의 복수의 싸이클로부터 기설정된 간격의 제3 싸이클에서, 상기 제2 연산 소자들 각각이, 상기 제1 연산 소자들 중 상기 동일한 컬럼에서 인접한 제1 연산 소자로부터 제공된 연산 데이터를 상기 기저장된 연산 데이터와 연산하도록 상기 제2 연산 소자들을 제어하여 상기 풀링을 수행하며,

상기 제1 싸이클 직후의 복수의 싸이클의 개수와 상기 제2 싸이클 직후의 복수의 싸이클의 개수는 동일한, 제어 방법.

청구항 14

제13항에 있어서,

상기 제1 연산 소자들을 제어하는 단계는,

상기 제1 싸이클 직후의 복수의 싸이클 중 최초 싸이클에서, 상기 제1 연산 소자들 각각에 인접한 제1 연산 소자가 제1 엘리먼트를 제공하도록 상기 제1 연산 소자들을 제어하며,

상기 제2 연산 소자들을 제어하는 단계는,

상기 제2 싸이클 직후의 복수의 싸이클 중 최초 싸이클에서, 상기 제2 연산 소자들 각각에 인접한 제2 연산 소자가 제2 엘리먼트를 제공하도록 상기 제2 연산 소자들을 제어하는, 제어 방법.

청구항 15

제14항에 있어서,

상기 제1 연산 소자들을 제어하는 단계는,

상기 제1 싸이클 직후의 복수의 싸이클 중 최초 싸이클 이후의 싸이클 동안, 상기 제1 연산 소자들 각각에 인접한 제1 연산 소자가 직전 싸이클에서의 연산 데이터를 제공하도록 상기 제1 연산 소자들을 제어하며,

상기 제2 연산 소자들을 제어하는 단계는,

상기 제2 싸이클 직후의 복수의 싸이클 중 최초 싸이클 이후의 싸이클 동안, 상기 제2 연산 소자들 각각에 인접한 제2 연산 소자가 직전 싸이클에서의 연산 데이터를 제공하도록 상기 제2 연산 소자들을 제어하는, 제어 방법.

청구항 16

제11항에 있어서,

상기 대상 데이터에 포함된 복수의 엘리먼트 중 제3 로우에 포함된 제3 엘리먼트들을 각각 상기 복수의 연산 소자 중 제3 로우에 배열된 제3 연산 소자들로 입력하는 단계; 및

상기 제3 연산 소자들 각각이, 인접한 제3 연산 소자로부터 제공된 데이터를 상기 입력된 제3 엘리먼트와 연산하도록 상기 제3 연산 소자들을 제어하는 단계;를 더 포함하며,

상기 풀링을 수행하는 단계는,

상기 제3 연산 소자들 각각이, 상기 제2 연산 소자들 중 동일한 컬럼에서 인접한 제2 연산 소자로부터 제공된 연산 데이터를 기저장된 연산 데이터와 연산하도록 상기 제3 연산 소자들을 제어하여 상기 풀링을 수행하며,

상기 인접한 제2 연산 소자로부터 제공된 연산 데이터는,

상기 인접한 제2 연산 소자와 동일한 컬럼에서 인접한 제1 연산 소자로부터 제공된 상기 연산 데이터와 상기 인접한 제2 연산 소자의 상기 기저장된 연산 데이터의 연산 결과인, 제어 방법.

청구항 17

제11항에 있어서,

상기 제1 연산 소자들을 제어하는 단계는,

상기 제1 연산 소자들 각각이, 상기 인접한 제1 연산 소자로부터 제공된 데이터를 상기 입력된 제1 엘리먼트와 덧셈 연산하도록 상기 제1 연산 소자들을 제어하고,

상기 제2 연산 소자들을 제어하는 단계는,

상기 제2 연산 소자들 각각이, 상기 인접한 제2 연산 소자로부터 제공된 데이터를 상기 입력된 제2 엘리먼트와 덧셈 연산하도록 상기 제2 연산 소자들을 제어하며,

상기 풀링을 수행하는 단계는,

상기 제2 연산 소자들 각각이, 상기 제1 연산 소자들 중 상기 동일한 컬럼에서 인접한 제1 연산 소자로부터 제공된 연산 데이터를 상기 기저장된 연산 데이터와 덧셈 연산하도록 상기 제2 연산 소자들을 제어하여 평균값(average) 풀링을 수행하는, 제어 방법.

청구항 18

제11항에 있어서,

상기 제1 연산 소자들을 제어하는 단계는,

상기 제1 연산 소자들 각각이, 상기 인접한 제1 연산 소자로부터 제공된 제1 엘리먼트를 상기 입력된 제1 엘리먼트와 비교 연산하도록 상기 제1 연산 소자들을 제어하고,

상기 제2 연산 소자들을 제어하는 단계는,

상기 제2 연산 소자들 각각이, 상기 인접한 제2 연산 소자로부터 제공된 제2 엘리먼트를 상기 입력된 제2 엘리먼트와 비교 연산하도록 상기 제2 연산 소자들을 제어하며,

상기 풀링을 수행하는 단계는,

상기 제2 연산 소자들 각각이, 상기 제1 연산 소자들 중 상기 동일한 컬럼에서 인접한 제1 연산 소자로부터 제공된 제1 엘리먼트를 기저장된 제2 엘리먼트와 비교 연산하도록 상기 제2 연산 소자들을 제어하여 최댓값(max) 풀링을 수행하는, 제어 방법.

청구항 19

제11항에 있어서,

상기 복수의 연산 소자 각각은,

제1 레지스터; 및

제2 레지스터;를 포함하고,

상기 제1 연산 소자들을 제어하는 단계는,

상기 제1 연산 소자들 각각이, 상기 인접한 제1 연산 소자의 제1 레지스터 또는 제2 레지스터로부터 제공된 데이터를 상기 제1 연산 소자들 각각의 제1 레지스터에 저장된 제1 엘리먼트와 연산하여 상기 제1 연산 소자들 각각의 제2 레지스터에 저장하도록 상기 제1 연산 소자들을 제어하고,

상기 제2 연산 소자들을 제어하는 단계는,

상기 제2 연산 소자들 각각이, 상기 인접한 제2 연산 소자의 제1 레지스터 또는 제2 레지스터로부터 제공된 데이터를 상기 제2 연산 소자들 각각의 제1 레지스터에 저장된 제2 엘리먼트와 연산하여 상기 제2 연산 소자들 각각의 제2 레지스터에 저장하도록 상기 제2 연산 소자들을 제어하는, 제어 방법.

청구항 20

제19항에 있어서,

상기 풀링을 수행하는 단계는,

상기 제2 연산 소자들 각각이, 상기 제1 연산 소자들 중 상기 동일한 컬럼에서 인접한 제1 연산 소자의 제1 레지스터로부터 제공된 연산 데이터를 상기 제2 연산 소자들 각각의 제1 레지스터에 저장된 상기 연산 데이터와 연산하여 상기 제2 연산 소자들 각각의 제2 레지스터에 저장하도록 상기 제2 연산 소자들을 제어하는, 제어 방법.

발명의 설명

기 술 분 야

[0001] 본 발명은 프로세서 및 그 제어 방법에 대한 것으로, 더욱 상세하게는 풀링(Pooling)을 수행하는 프로세서 및 그 제어 방법에 대한 것이다.

배 경 기 술

[0002] 딥러닝은 인공 신경망에 기반을 둔 머신 러닝 기술의 한 종류로, 인공 신경망이 다층 구조로 설계되어 깊어지더

라도 학습을 위한 데이터들을 비지도 학습(Unsupervised Learning)을 활용한 전처리나, 데이터를 여러 층 너머로 한번에 전달함으로써 학습 효율을 향상시킬 수 있다. 특히, 딥러닝은 인터넷에 의한 빅데이터 및 이를 처리하기 위한 컴퓨팅 능력의 향상으로 최근 비약적인 발전을 보이고 있다.

[0003] 이중, 컨볼루션 신경망(Convolutional Neural Network, CNN)은 2차원 데이터의 학습에 적합한 구조를 가지고 있으며, 컨볼루션(convolution) 연산, 풀링(Pooling) 연산 등을 포함할 수 있다.

[0004] 여기서, 풀링은 로컬(local) 영역 내의 의미 있는 시그널(Signal)을 검출하기 위해 이용된다. 예를 들어, 이미지에서 눈동자를 찾기 위한 필터로 컨볼루션을 수행하면, 결과값으로 눈동자 위치에 높은 수치가 기록되나, 이러한 수치는 방대한 결과값 중 일부이므로 효과적으로 다음 레이어로 전달되지 않을 수 있다. 이 경우, 최댓값 풀링(Max Pooling)을 이용하여, 도 1a와 같이 로컬 영역 내 가장 큰 시그널을 다음 레이어로 전달할 수 있다. 그리고, 다음 레이어로 전달된 강한 시그널은 필터 학습시 역전파(Back-propagation) 알고리즘을 통하여 다시 앞으로 전달되어, 눈동자를 검출하는 필터를 더 효과적으로 학습시킬 수 있다. 따라서, 풀링은 추론(Inference)과 학습 효율을 높이는 중요한 기능을 수행한다.

[0005] 또는, 풀링은 도 1b와 같이 Feature Map 데이터의 축소를 위하여 사용될 수 있다. 풀링을 수행하지 않는 경우 컨볼루션이 진행됨에 따라 뎁스(depth) 방향(Z축 방향)으로 크기가 증가할 수 있으나, 풀링을 수행하는 경우 X축 및 Y축 방향의 크기를 감소시켜 연산량 및 메모리 요구량을 감소시킬 수 있다.

[0006] 다만, 종래에는 풀링 연산을 위해 1차원 계산기 구조(1D Array)가 이용되었다. 이 경우, 도 1c에 도시된 바와 같이, 컨볼루션을 위한 2차원 계산기 구조(2D Array)와 풀링을 위한 1D Array가 별도의 하드웨어로 구현되어야 하며, 이는 하드웨어 면적 및 전력 소모와 같은 비용을 증가시키는 문제가 있다. 또한, 도 1d에 도시된 바와 같이, 하드웨어 자원의 Load-balancing 문제를 발생시키기도 한다.

[0007] 또한, 1D Array 구조를 이용한 풀링의 경우, SIMD(Single Instruction Multiple Data)로 불리는 방법을 이용하여, 하나의 명령어(Instruction)로 다수의 연산 소자(Processing Element, PE)를 동시에 제어하여 연산을 수행한다. 이 경우, 하나의 연산 소자는 Feature Map 데이터의 하나의 뎁스에 포함된 데이터를 처리하게 되며, 각 연산 소자는 하나의 뎁스에 포함된 데이터의 로컬 영역 별로 풀링을 수행하게 된다.

[0008] 예를 들어, 로컬 영역의 크기가 3 × 3이고, 스트라이드(Stride) 1로 풀링을 수행하는 경우, 도 1e에 도시된 바와 같이, 첫 번째 연산 소자는 4, 2, 4, 7, 3, 2, 4, 3, 8을 순차적으로 입력받아 풀링을 수행하고, 2, 4, 3, 3, 2, 6, 3, 8, 4를 순차적으로 입력받아 풀링을 수행하게 된다. 이 경우, 도 1e의 8은 로컬 영역이 변경됨에 따라 각 영역 별로 9번 메모리 read 후, 9번 add 연산을 수행하게 된다. 즉, 불필요한 메모리 접근과 연산량이 증가하는 문제점이 있다.

[0009] 또는, 도 1f에 도시된 바와 같이, 풀링 뿐만 아니라 컨볼루션도 SIMD 방식을 이용할 수도 있으나, 컨볼루션의 경우 데이터 재사용(Data reuse)이 불가한 단점이 있다.

[0010] 이상과 같이, 1D Array 구조를 이용한 풀링의 경우, 다양한 문제가 발생하게 된다.

발명의 내용

해결하려는 과제

[0011] 본 발명은 상술한 필요성에 따른 것으로, 본 발명의 목적은 2D Array 구조의 복수의 연산 소자(Processing Element)를 이용하여 풀링(Pooling)을 수행하는 프로세서 및 그 제어 방법을 제공함에 있다.

과제의 해결 수단

[0012] 이상과 같은 목적을 달성하기 위한 본 발명의 일 실시 예에 따르면, 프로세서는 매트릭스 형태로 배열된 복수의 연산 소자(Processing Element) 및 복수의 싸이클 동안 상기 복수의 연산 소자 유닛를 제어하여 대상 데이터를 처리하는 제어부(controller)를 포함하며, 상기 제어부는 상기 대상 데이터에 포함된 복수의 엘리먼트 중 제1 로우(row)에 포함된 제1 엘리먼트들을 각각 상기 복수의 연산 소자 중 제1 로우에 배열된 제1 연산 소자들로 입력하며, 상기 제1 연산 소자들 각각이, 인접한 제1 연산 소자로부터 제공된 데이터를 상기 입력된 제1 엘리먼트와 연산하도록 상기 제1 연산 소자들을 제어하고, 상기 복수의 엘리먼트 중 제2 로우에 포함된 제2 엘리먼트들을 각각 상기 복수의 연산 소자 중 제2 로우에 배열된 제2 연산 소자들로 입력하며, 상기 제2 연산 소자들 각각이, 인접한 제2 연산 소자로부터 제공된 데이터를 상기 입력된 제2 엘리먼트와 연산하도록 상기 제2 연산 소자

들을 제어하고, 상기 제2 연산 소자들 각각이, 상기 제1 연산 소자들 중 동일한 컬럼(column)에서 인접한 제1 연산 소자로부터 제공된 연산 데이터를 기저장된 연산 데이터와 연산하도록 상기 제2 연산 소자들을 제어하여 상기 대상 데이터에 대한 풀링(Pooling) 결과를 획득할 수 있다.

[0013] 또한, 상기 제어부는 제1 싸이클에서, 상기 제1 엘리먼트들을 각각 상기 제1 연산 소자들로 입력하며, 상기 제1 싸이클 직후의 제2 싸이클에서, 상기 제1 연산 소자들 각각이, 상기 인접한 제1 연산 소자로부터 제공된 데이터를 상기 입력된 제1 엘리먼트와 연산하도록 상기 제1 연산 소자들을 제어하고, 상기 제2 엘리먼트들을 각각 상기 제2 연산 소자들로 입력하며, 상기 제2 싸이클 직후의 제3 싸이클에서, 상기 제2 연산 소자들 각각이, 상기 인접한 제2 연산 소자로부터 제공된 데이터를 상기 입력된 제2 엘리먼트와 연산하도록 상기 제2 연산 소자들을 제어하고, 상기 제3 싸이클 직후의 제4 싸이클에서, 상기 제2 연산 소자들 각각이, 상기 제1 연산 소자들 중 상기 동일한 컬럼에서 인접한 제1 연산 소자로부터 제공된 연산 데이터를 상기 기저장된 연산 데이터와 연산하도록 상기 제2 연산 소자들을 제어하여 상기 풀링 결과를 획득할 수 있다.

[0014] 그리고, 상기 제어부는 제1 싸이클에서, 상기 제1 엘리먼트들을 각각 상기 제1 연산 소자들로 입력하며, 상기 제1 싸이클 직후의 복수의 싸이클 동안, 상기 제1 연산 소자들 각각이, 상기 인접한 제1 연산 소자로부터 제공된 데이터를 상기 입력된 제1 엘리먼트와 연산하도록 상기 제1 연산 소자들을 제어하고, 상기 복수의 싸이클 중 하나인 제2 싸이클에서, 상기 제2 엘리먼트들을 각각 상기 제2 연산 소자들로 입력하며, 상기 제2 싸이클 직후의 복수의 싸이클 동안, 상기 제2 연산 소자들 각각이, 상기 인접한 제2 연산 소자로부터 제공된 데이터를 상기 입력된 제2 엘리먼트와 연산하도록 상기 제2 연산 소자들을 제어하고, 상기 제2 싸이클 직후의 복수의 싸이클로부터 기설정된 간격의 제3 싸이클에서, 상기 제2 연산 소자들 각각이, 상기 제1 연산 소자들 중 상기 동일한 컬럼에서 인접한 제1 연산 소자로부터 제공된 연산 데이터를 상기 기저장된 연산 데이터와 연산하도록 상기 제2 연산 소자들을 제어하여 상기 풀링 결과를 획득하며, 상기 제1 싸이클 직후의 복수의 싸이클의 개수와 상기 제2 싸이클 직후의 복수의 싸이클의 개수는 동일할 수 있다.

[0015] 또한, 상기 제어부는 상기 제1 싸이클 직후의 복수의 싸이클 중 최초 싸이클에서, 상기 제1 연산 소자들 각각에 인접한 제1 연산 소자가 제1 엘리먼트를 제공하도록 상기 제1 연산 소자들을 제어하며, 상기 제2 싸이클 직후의 복수의 싸이클 중 최초 싸이클에서, 상기 제2 연산 소자들 각각에 인접한 제2 연산 소자가 제2 엘리먼트를 제공하도록 상기 제2 연산 소자들을 제어할 수 있다.

[0016] 그리고, 상기 제어부는 상기 제1 싸이클 직후의 복수의 싸이클 중 최초 싸이클 이후의 싸이클 동안, 상기 제1 연산 소자들 각각에 인접한 제1 연산 소자가 직전 싸이클에서의 연산 데이터를 제공하도록 상기 제1 연산 소자들을 제어하며, 상기 제2 싸이클 직후의 복수의 싸이클 중 최초 싸이클 이후의 싸이클 동안, 상기 제2 연산 소자들 각각에 인접한 제2 연산 소자가 직전 싸이클에서의 연산 데이터를 제공하도록 상기 제2 연산 소자들을 제어할 수 있다.

[0017] 또한, 상기 제어부는 상기 대상 데이터에 포함된 복수의 엘리먼트 중 제3 로우에 포함된 제3 엘리먼트들을 각각 상기 복수의 연산 소자 중 제3 로우에 배열된 제3 연산 소자들로 입력하며, 상기 제3 연산 소자들 각각이, 인접한 제3 연산 소자로부터 제공된 데이터를 상기 입력된 제3 엘리먼트와 연산하도록 상기 제3 연산 소자들을 제어하고, 상기 제3 연산 소자들 각각이, 상기 제2 연산 소자들 중 동일한 컬럼에서 인접한 제2 연산 소자로부터 제공된 연산 데이터를 기저장된 연산 데이터와 연산하도록 상기 제3 연산 소자들을 제어하여 상기 풀링 결과를 획득하며, 상기 인접한 제2 연산 소자로부터 제공된 연산 데이터는 상기 인접한 제2 연산 소자와 동일한 컬럼에서 인접한 제1 연산 소자로부터 제공된 상기 연산 데이터와 상기 인접한 제2 연산 소자의 상기 기저장된 연산 데이터의 연산 결과일 수 있다.

[0018] 그리고, 상기 제어부는 상기 제1 연산 소자들 각각이, 상기 인접한 제1 연산 소자로부터 제공된 데이터를 상기 입력된 제1 엘리먼트와 덧셈 연산하도록 상기 제1 연산 소자들을 제어하고, 상기 제2 연산 소자들 각각이, 상기 인접한 제2 연산 소자로부터 제공된 데이터를 상기 입력된 제2 엘리먼트와 덧셈 연산하도록 상기 제2 연산 소자들을 제어하며, 상기 제2 연산 소자들 각각이, 상기 제1 연산 소자들 중 상기 동일한 컬럼에서 인접한 제1 연산 소자로부터 제공된 연산 데이터를 상기 기저장된 연산 데이터와 덧셈 연산하도록 상기 제2 연산 소자들을 제어하여 평균값(average) 풀링 결과를 획득할 수 있다.

[0019] 또한, 상기 제어부는 상기 제1 연산 소자들 각각이, 상기 인접한 제1 연산 소자로부터 제공된 제1 엘리먼트를 상기 입력된 제1 엘리먼트와 비교 연산하도록 상기 제1 연산 소자들을 제어하고, 상기 제2 연산 소자들 각각이, 상기 인접한 제2 연산 소자로부터 제공된 제2 엘리먼트를 상기 입력된 제2 엘리먼트와 비교 연산하도록 상기 제2 연산 소자들을 제어하며, 상기 제2 연산 소자들 각각이, 상기 제1 연산 소자들 중 상기 동일한 컬럼에서 인접

한 제1 연산 소자로부터 제공된 제1 엘리먼트를 기저장된 제2 엘리먼트와 비교 연산하도록 상기 제2 연산 소자들을 제어하여 최댓값(max) 풀링 결과를 획득할 수 있다.

[0020] 그리고, 상기 복수의 연산 소자 각각은 제1 레지스터 및 제2 레지스터를 포함하고, 상기 제어부는 상기 제1 연산 소자들 각각이, 상기 인접한 제1 연산 소자의 제1 레지스터 또는 제2 레지스터로부터 제공된 데이터를 상기 제1 연산 소자들 각각의 제1 레지스터에 저장된 제1 엘리먼트와 연산하여 상기 제1 연산 소자들 각각의 제2 레지스터에 저장하도록 상기 제1 연산 소자들을 제어하고, 상기 제2 연산 소자들 각각이, 상기 인접한 제2 연산 소자의 제1 레지스터 또는 제2 레지스터로부터 제공된 데이터를 상기 제2 연산 소자들 각각의 제1 레지스터에 저장된 제2 엘리먼트와 연산하여 상기 제2 연산 소자들 각각의 제2 레지스터에 저장하도록 상기 제2 연산 소자들을 제어할 수 있다.

[0021] 또한, 상기 제어부는 상기 제2 연산 소자들 각각이, 상기 제1 연산 소자들 중 상기 동일한 컬럼에서 인접한 제1 연산 소자의 제1 레지스터로부터 제공된 연산 데이터를 상기 제2 연산 소자들 각각의 제1 레지스터에 저장된 상기 연산 데이터와 연산하여 상기 제2 연산 소자들 각각의 제2 레지스터에 저장하도록 상기 제2 연산 소자들을 제어할 수 있다.

[0022] 한편, 본 발명의 일 실시 예에 따르면, 매트릭스 형태로 배열된 복수의 연산 소자(Processing Element)를 포함하며, 복수의 싸이클 동안 상기 복수의 연산 소자를 이용하여 대상 데이터에 대한 풀링(pooling)을 수행하는 프로세서의 제어 방법은 상기 대상 데이터에 포함된 복수의 엘리먼트 중 제1 로우(row)에 포함된 제1 엘리먼트들을 각각 상기 복수의 연산 소자 중 제1 로우에 배열된 제1 연산 소자들로 입력하는 단계, 상기 제1 연산 소자들 각각이, 인접한 제1 연산 소자로부터 제공된 데이터를 상기 입력된 제1 엘리먼트와 연산하도록 상기 제1 연산 소자들을 제어하고, 상기 복수의 엘리먼트 중 제2 로우에 포함된 제2 엘리먼트들을 각각 상기 복수의 연산 소자 중 제2 로우에 배열된 제2 연산 소자들로 입력하는 단계, 상기 제2 연산 소자들 각각이, 인접한 제2 연산 소자로부터 제공된 데이터를 상기 입력된 제2 엘리먼트와 연산하도록 상기 제2 연산 소자들을 제어하는 단계 및 상기 제2 연산 소자들 각각이, 상기 제1 연산 소자들 중 동일한 컬럼(column)에서 인접한 제1 연산 소자로부터 제공된 연산 데이터를 기저장된 연산 데이터와 연산하도록 상기 제2 연산 소자들을 제어하여 상기 풀링을 수행하는 단계를 포함한다.

[0023] 또한, 상기 제1 연산 소자들로 입력하는 단계는 제1 싸이클에서, 상기 제1 엘리먼트들을 각각 상기 제1 연산 소자들로 입력하며, 상기 제2 연산 소자들로 입력하는 단계는 상기 제1 싸이클 직후의 제2 싸이클에서, 상기 제1 연산 소자들 각각이, 상기 인접한 제1 연산 소자로부터 제공된 데이터를 상기 입력된 제1 엘리먼트와 연산하도록 상기 제1 연산 소자들을 제어하고, 상기 제2 엘리먼트들을 각각 상기 제2 연산 소자들로 입력하며, 상기 제2 연산 소자들을 제어하는 단계는 상기 제2 싸이클 직후의 제3 싸이클에서, 상기 제2 연산 소자들 각각이, 상기 인접한 제2 연산 소자로부터 제공된 데이터를 상기 입력된 제2 엘리먼트와 연산하도록 상기 제2 연산 소자들을 제어하고, 상기 풀링을 수행하는 단계는 상기 제3 싸이클 직후의 제4 싸이클에서, 상기 제2 연산 소자들 각각이, 상기 제1 연산 소자들 중 상기 동일한 컬럼에서 인접한 제1 연산 소자로부터 제공된 연산 데이터를 상기 기저장된 연산 데이터와 연산하도록 상기 제2 연산 소자들을 제어하여 상기 풀링을 수행할 수 있다.

[0024] 그리고, 상기 제1 연산 소자들로 입력하는 단계는 제1 싸이클에서, 상기 제1 엘리먼트들을 각각 상기 제1 연산 소자들로 입력하며, 상기 제2 연산 소자들로 입력하는 단계는 상기 제1 싸이클 직후의 복수의 싸이클 동안, 상기 제1 연산 소자들 각각이, 상기 인접한 제1 연산 소자로부터 제공된 데이터를 상기 입력된 제1 엘리먼트와 연산하도록 상기 제1 연산 소자들을 제어하고, 상기 복수의 싸이클 중 하나인 제2 싸이클에서, 상기 제2 엘리먼트들을 각각 상기 제2 연산 소자들로 입력하며, 상기 제2 연산 소자들을 제어하는 단계는 상기 제2 싸이클 직후의 복수의 싸이클 동안, 상기 제2 연산 소자들 각각이, 상기 인접한 제2 연산 소자로부터 제공된 데이터를 상기 입력된 제2 엘리먼트와 연산하도록 상기 제2 연산 소자들을 제어하고, 상기 풀링을 수행하는 단계는 상기 제2 싸이클 직후의 복수의 싸이클로부터 기설정된 간격의 제3 싸이클에서, 상기 제2 연산 소자들 각각이, 상기 제1 연산 소자들 중 상기 동일한 컬럼에서 인접한 제1 연산 소자로부터 제공된 연산 데이터를 상기 기저장된 연산 데이터와 연산하도록 상기 제2 연산 소자들을 제어하여 상기 풀링을 수행하며, 상기 제1 싸이클 직후의 복수의 싸이클의 개수와 상기 제2 싸이클 직후의 복수의 싸이클의 개수는 동일할 수 있다.

[0025] 또한, 상기 제1 연산 소자들을 제어하는 단계는 상기 제1 싸이클 직후의 복수의 싸이클 중 최초 싸이클에서, 상기 제1 연산 소자들 각각에 인접한 제1 연산 소자가 제1 엘리먼트를 제공하도록 상기 제1 연산 소자들을 제어하며, 상기 제2 연산 소자들을 제어하는 단계는 상기 제2 싸이클 직후의 복수의 싸이클 중 최초 싸이클에서, 상기 제2 연산 소자들 각각에 인접한 제2 연산 소자가 제2 엘리먼트를 제공하도록 상기 제2 연산 소자들을 제어할 수

있다.

[0026] 그리고, 상기 제1 연산 소자들을 제어하는 단계는 상기 제1 싸이클 직후의 복수의 싸이클 중 최초 싸이클 이후의 싸이클 동안, 상기 제1 연산 소자들 각각에 인접한 제1 연산 소자가 직전 싸이클에서의 연산 데이터를 제공하도록 상기 제1 연산 소자들을 제어하며, 상기 제2 연산 소자들을 제어하는 단계는 상기 제2 싸이클 직후의 복수의 싸이클 중 최초 싸이클 이후의 싸이클 동안, 상기 제2 연산 소자들 각각에 인접한 제2 연산 소자가 직전 싸이클에서의 연산 데이터를 제공하도록 상기 제2 연산 소자들을 제어할 수 있다.

[0027] 또한, 상기 대상 데이터에 포함된 복수의 엘리먼트 중 제3 로우에 포함된 제3 엘리먼트들을 각각 상기 복수의 연산 소자 중 제3 로우에 배열된 제3 연산 소자들로 입력하는 단계 및 상기 제3 연산 소자들 각각이, 인접한 제3 연산 소자로부터 제공된 데이터를 상기 입력된 제3 엘리먼트와 연산하도록 상기 제3 연산 소자들을 제어하는 단계를 더 포함하며, 상기 풀링을 수행하는 단계는 상기 제3 연산 소자들 각각이, 상기 제2 연산 소자들 중 동일한 컬럼에서 인접한 제2 연산 소자로부터 제공된 연산 데이터를 기저장된 연산 데이터와 연산하도록 상기 제3 연산 소자들을 제어하여 상기 풀링을 수행하며, 상기 인접한 제2 연산 소자로부터 제공된 연산 데이터는 상기 인접한 제2 연산 소자와 동일한 컬럼에서 인접한 제1 연산 소자로부터 제공된 상기 연산 데이터와 상기 인접한 제2 연산 소자의 상기 기저장된 연산 데이터의 연산 결과일 수 있다.

[0028] 그리고, 상기 제1 연산 소자들을 제어하는 단계는 상기 제1 연산 소자들 각각이, 상기 인접한 제1 연산 소자로부터 제공된 데이터를 상기 입력된 제1 엘리먼트와 덧셈 연산하도록 상기 제1 연산 소자들을 제어하고, 상기 제2 연산 소자들을 제어하는 단계는 상기 제2 연산 소자들 각각이, 상기 인접한 제2 연산 소자로부터 제공된 데이터를 상기 입력된 제2 엘리먼트와 덧셈 연산하도록 상기 제2 연산 소자들을 제어하며, 상기 풀링을 수행하는 단계는 상기 제2 연산 소자들 각각이, 상기 제1 연산 소자들 중 상기 동일한 컬럼에서 인접한 제1 연산 소자로부터 제공된 연산 데이터를 상기 기저장된 연산 데이터와 덧셈 연산하도록 상기 제2 연산 소자들을 제어하여 평균값(average) 풀링을 수행할 수 있다.

[0029] 또한, 상기 제1 연산 소자들을 제어하는 단계는 상기 제1 연산 소자들 각각이, 상기 인접한 제1 연산 소자로부터 제공된 제1 엘리먼트를 상기 입력된 제1 엘리먼트와 비교 연산하도록 상기 제1 연산 소자들을 제어하고, 상기 제2 연산 소자들을 제어하는 단계는 상기 제2 연산 소자들 각각이, 상기 인접한 제2 연산 소자로부터 제공된 제2 엘리먼트를 상기 입력된 제2 엘리먼트와 비교 연산하도록 상기 제2 연산 소자들을 제어하며, 상기 풀링을 수행하는 단계는 상기 제2 연산 소자들 각각이, 상기 제1 연산 소자들 중 상기 동일한 컬럼에서 인접한 제1 연산 소자로부터 제공된 제1 엘리먼트를 기저장된 제2 엘리먼트와 비교 연산하도록 상기 제2 연산 소자들을 제어하여 최댓값(max) 풀링을 수행할 수 있다.

[0030] 그리고, 상기 복수의 연산 소자 각각은 제1 레지스터 및 제2 레지스터를 포함하고, 상기 제1 연산 소자들을 제어하는 단계는 상기 제1 연산 소자들 각각이, 상기 인접한 제1 연산 소자의 제1 레지스터 또는 제2 레지스터로부터 제공된 데이터를 상기 제1 연산 소자들 각각의 제1 레지스터에 저장된 제1 엘리먼트와 연산하여 상기 제1 연산 소자들 각각의 제2 레지스터에 저장하도록 상기 제1 연산 소자들을 제어하고, 상기 제2 연산 소자들을 제어하는 단계는 상기 제2 연산 소자들 각각이, 상기 인접한 제2 연산 소자의 제1 레지스터 또는 제2 레지스터로부터 제공된 데이터를 상기 제2 연산 소자들 각각의 제1 레지스터에 저장된 제2 엘리먼트와 연산하여 상기 제2 연산 소자들 각각의 제2 레지스터에 저장하도록 상기 제2 연산 소자들을 제어할 수 있다.

[0031] 또한, 상기 풀링을 수행하는 단계는 상기 제2 연산 소자들 각각이, 상기 제1 연산 소자들 중 상기 동일한 컬럼에서 인접한 제1 연산 소자의 제1 레지스터로부터 제공된 연산 데이터를 상기 제2 연산 소자들 각각의 제1 레지스터에 저장된 상기 연산 데이터와 연산하여 상기 제2 연산 소자들 각각의 제2 레지스터에 저장하도록 상기 제2 연산 소자들을 제어할 수 있다.

발명의 효과

[0032] 이상과 같은 본 발명의 다양한 실시 예에 따르면, 프로세서는 2D Array 구조의 복수의 연산 소자(Processing Element)를 이용하여 컨볼루션 뿐만 아니라 풀링을 수행하게 되어, 하드웨어 이원화 문제, Load-balancing 문제 및 중복 연산 문제를 해결하고, 피크 대역폭(Peak bandwidth)을 낮출 수 있다.

도면의 간단한 설명

[0033] 도 1a 내지 도 1f는 종래 기술에 따른 풀링(Pooling)을 설명하기 위한 도면들이다.

도 2a는 본 발명의 일 실시 예에 따른 프로세서의 구성을 나타내는 블럭도이다.

도 2b는 본 개시의 일 실시 예에 따른 프로세서의 세부 구성을 나타내는 도면이다.

도 2c 및 도 2d는 본 개시의 일 실시 예에 따른 복수의 연산 소자 각각의 세부 구성 및 연결 상태를 나타내는 도면들이다.

도 3 내지 도 8은 본 개시의 일 실시 예에 따른 복수의 연산 소자(110)의 동작을 설명하기 위한 도면들이다.

도 9a 내지 도 11c는 본 개시의 다양한 실시 예에 따른 싸이클 별 제어부의 동작 방법을 나타내는 도면들이다.

도 12는 본 개시의 일 실시 예에 따른 복수의 연산 소자의 Utilization을 높이는 방법을 설명하기 위한 도면이다.

도 13은 본 개시의 일 실시 예에 따른 명령어의 로테이션(rotation)을 설명하기 위한 도면이다.

도 14는 본 개시의 일 실시 예에 따른 효과를 설명하기 위한 도면이다.

도 15는 본 개시의 일 실시 예에 따른 프로세서를 이용하는 전자 장치를 설명하기 위한 도면이다.

도 16은 본 개시의 일 실시 예에 따른 프로세서의 제어 방법을 설명하기 위한 흐름도이다.

발명을 실시하기 위한 구체적인 내용

[0034] 이하에서, 첨부된 도면을 이용하여 본 발명의 다양한 실시 예들에 대하여 구체적으로 설명한다.

[0035] 도 2a는 본 발명의 일 실시 예에 따른 프로세서(100)의 구성을 나타내는 블럭도이다.

[0036] 도 2a에 도시된 바와 같이, 프로세서(100)는 복수의 연산 소자(Processing Element, 110) 및 제어부(controller, 120)를 포함한다.

[0037] 프로세서(100)는 풀링(Pooling)을 수행하는 장치일 수 있다. 예를 들어, 프로세서(100)는 메모리에 저장된 대상 데이터를 3차원 형태로 식별하고, 3차원 형태의 대상 데이터를 X축, Y축 방향을 기준으로 풀링을 수행할 수 있다. 그리고, 프로세서(100)는 X축, Y축 방향의 기설정된 크기의 로컬 영역 내에서 최댓값을 출력하는 방식으로 최댓값 풀링(Max Pooling)을 수행할 수 있다. 또는, 프로세서(100)는 X축, Y축 방향의 기설정된 크기의 로컬 영역 내의 평균값을 출력하는 방식으로 평균값 풀링(Average Pooling)을 수행할 수 있다. 여기서, 기설정된 크기는 풀링의 단위가 되는 영역의 크기를 의미하며, 커널 사이즈(Kernel size)로도 불린다. 대상 데이터는 풀링의 대상이 되는 데이터로서, 인공지능 모델을 적용하기 위한 초기 데이터(ex, 이미지) 또는 인공지능 모델의 연산 과정에서 획득되는 Feature Map일 수 있다.

[0038] 한편, 프로세서(100)는 로컬 영역의 오버랩 정도를 나타내는 스트라이드(Stride)에 기초하여 복수의 로컬 영역 각각에 대한 풀링을 수행할 수 있다. 설명의 편의를 위해 도 1e의 도면을 통해 설명하면, 도 1e는 스트라이드가 1인 경우의 9개의 로컬 영역에 대한 풀링을 나타낸다. 스트라이드가 2인 경우, 프로세서(100)는 도 1e의 9개의 로컬 영역 중 꼭짓점에 위치한 4개의 로컬 영역에 대한 풀링만을 수행하게 된다. 즉, 스트라이드는 현재 로컬 영역과 다음 로컬 영역의 거리 차이라고 볼 수 있다.

[0039] 또한, 프로세서(100)는 컨볼루션(Convolution)을 수행할 수도 있다.

[0040] 복수의 연산 소자(110)는 매트릭스 형태로 배열될 수 있고, 인접한 연산 소자 간에는 데이터의 일방향 시프트 또는 양방향 시프트가 가능하다.

[0041] 복수의 연산 소자(110) 각각은 풀링을 수행하기 위한 가산기(Adder), 멀티플렉서(Multiplexer) 및 레지스터(Register) 등을 포함할 수 있다. 다만, 이에 한정되는 것은 아니며, 복수의 연산 소자(110) 각각은 컨볼루션을 수행하기 위한 곱셈기(multiplier) 및 기타 산술 논리 연산 장치(Arithmetic Logic Unit, ALU)를 더 포함할 수도 있다.

[0042] 여기서, 가산기는 복수의 데이터를 입력받고, 이들의 합을 출력하는 회로이고, 멀티플렉서는 복수의 데이터를 입력받고, 이중 하나의 데이터를 출력하는 회로이다. 레지스터는 극히 소량의 데이터나 처리 중인 중간 결과를 일시적으로 기억해 두는 고속의 데이터 저장 매체이며, 곱셈기는 복수의 데이터를 입력받고, 이들의 곱을 출력하는 회로이다.

[0043] 복수의 연산 소자(110) 각각은 제어부(120)의 제어에 따라 풀링을 수행하는데 이용될 수 있다. 또한, 복수의 연

산 소자(110) 각각은 제어부(120)의 제어에 따라 컨볼루션을 수행하는데 이용될 수도 있다.

[0044] 제어부(120)는 프로세서(100)의 동작을 전반적으로 제어한다.

[0045] 제어부(120)는 복수의 싸이클 동안 복수의 연산 소자(110)를 제어하여 대상 데이터를 처리할 수 있다.

[0046] 제어부(120)는 프로세서(100)의 외부에 구비된 메모리로부터 대상 데이터 중 일부를 독출하여 복수의 연산 소자(110)에 입력할 수 있다. 예를 들어, 제어부(120)는 대상 데이터에 포함된 복수의 엘리먼트 중 제1 로우(row)에 포함된 제1 엘리먼트들을 각각 복수의 연산 소자(110) 중 제1 로우에 배열된 제1 연산 소자들로 입력할 수 있다. 여기서, 제1 로우에 포함된 제1 엘리먼트들은 대상 데이터의 복수의 로우 중 하나에 포함된 데이터들을 의미하며, 제1 로우에 배열된 제1 연산 소자들은 복수의 연산 소자(110)의 복수의 로우 중 하나에 포함된 연산 소자들을 의미한다. 그리고, 제1 로우에 포함된 제1 엘리먼트들의 개수는 복수의 연산 소자(110)의 열(column)의 개수와 동일할 수 있다. 예를 들어, 복수의 연산 소자(110)가 5 × 5의 매트릭스 형태인 경우, 제1 로우에 포함된 엘리먼트들의 개수가 5개를 초과하더라도, 제어부(120)는 5개의 제1 엘리먼트들만을 독출하여 각각 제1 연산 소자들로 입력할 수 있다. 또한, 제1 엘리먼트들은 대상 데이터의 제1 로우에서 연속된 데이터일 수 있다. 즉, 제어부(120)는 대상 데이터에 포함된 복수의 엘리먼트 중 일부인 제1 엘리먼트들을 그룹핑하고, 그룹핑된 제1 엘리먼트들을 동일하게 처리할 수 있다. 이러한 동작을 통해 병렬적인 연산이 가능하며, 이에 대한 구체적인 설명은 후술한다.

[0047] 그리고, 제어부(120)는 제1 연산 소자들 각각이, 인접한 제1 연산 소자로부터 제공된 데이터를 입력된 제1 엘리먼트와 연산하도록 제1 연산 소자들을 제어할 수 있다. 예를 들어, 제어부(120)는 제1 연산 소자들 각각이, 우측으로 인접한 제1 연산 소자로 메모리로부터 입력된 제1 엘리먼트를 전송하도록 제1 연산 소자들을 제어할 수 있다. 그리고, 제어부(120)는 제1 연산 소자들 각각이, 좌측으로 인접한 제1 연산 소자로부터 입력된 제1 엘리먼트와 메모리로부터 입력된 제1 엘리먼트를 연산하도록 제1 연산 소자들을 제어할 수 있다. 제어부(120)는 이러한 과정을 복수의 싸이클 동안 수행할 수 있고, 이 경우 제1 연산 소자들 각각이 전송하는 데이터는 제1 엘리먼트가 아닐 수 있으며, 이에 대한 구체적인 설명은 도면을 통해 후술한다.

[0048] 제어부(120)는 복수의 엘리먼트 중 제2 로우에 포함된 제2 엘리먼트들을 각각 복수의 연산 소자(110) 중 제2 로우에 배열된 제2 연산 소자들로 입력할 수 있다. 이러한 동작은 제1 연산 소자들로 제1 엘리먼트들을 입력하는 동작과 동일하므로 구체적인 설명은 생략한다.

[0049] 제2 연산 소자들로 제2 엘리먼트들을 입력하는 싸이클은 제1 연산 소자들 각각이 연산을 수행하는 적어도 하나의 싸이클 중 하나일 수 있다.

[0050] 제어부(120)는 제2 연산 소자들 각각이, 인접한 제2 연산 소자로부터 제공된 데이터를 입력된 제2 엘리먼트와 연산하도록 제2 연산 소자들을 제어할 수 있다. 이러한 동작은 제1 연산 소자들 각각이 연산을 수행하는 동작과 동일하므로 구체적인 설명은 생략한다.

[0051] 제어부(120)는 제2 연산 소자들 각각이, 제1 연산 소자들 중 동일한 컬럼(column)에서 인접한 제1 연산 소자로부터 제공된 연산 데이터를 기저장된 연산 데이터와 연산하도록 제2 연산 소자들을 제어하여 대상 데이터에 대한 풀링 결과를 획득할 수 있다.

[0052] 한편, 제어부(120)는 제1 싸이클에서, 제1 엘리먼트들을 각각 제1 연산 소자들로 입력하며, 제1 싸이클 직후의 제2 싸이클에서, 제1 연산 소자들 각각이, 인접한 제1 연산 소자로부터 제공된 데이터를 입력된 제1 엘리먼트와 연산하도록 제1 연산 소자들을 제어하고, 제2 엘리먼트들을 제2 연산 소자들로 입력하며, 제2 싸이클 직후의 제3 싸이클에서, 제2 연산 소자들 각각이, 인접한 제2 연산 소자로부터 제공된 데이터를 입력된 제2 엘리먼트와 연산하도록 제2 연산 소자들을 제어하고, 제3 싸이클 직후의 제4 싸이클에서, 제2 연산 소자들 각각이, 제1 연산 소자들 중 동일한 컬럼에서 인접한 제1 연산 소자로부터 제공된 연산 데이터를 기저장된 연산 데이터와 연산하도록 제2 연산 소자들을 제어하여 풀링 결과를 획득할 수 있다.

[0053] 또는, 제어부(120)는 제1 싸이클에서, 제1 엘리먼트들을 각각 제1 연산 소자들로 입력하며, 제1 싸이클 직후의 복수의 싸이클 동안, 제1 연산 소자들 각각이, 인접한 제1 연산 소자로부터 제공된 데이터를 입력된 제1 엘리먼트와 연산하도록 제1 연산 소자들을 제어하고, 복수의 싸이클 중 하나인 제2 싸이클에서, 제2 엘리먼트들을 각각 제2 연산 소자들로 입력하며, 제2 싸이클 직후의 복수의 싸이클 동안, 제2 연산 소자들 각각이, 인접한 제2 연산 소자로부터 제공된 데이터를 입력된 제2 엘리먼트와 연산하도록 제2 연산 소자들을 제어하고, 제2 싸이클 직후의 복수의 싸이클로부터 기설정된 간격의 제3 싸이클에서, 제2 연산 소자들 각각이, 제1 연산 소자들 중 동일한 컬럼에서 인접한 제1 연산 소자로부터 제공된 연산 데이터를 기저장된 연산 데이터와 연산하도록 제2 연산

소자들을 제어하여 풀링 결과를 획득할 수 있다. 여기서, 제1 싸이클 직후의 복수의 싸이클의 개수와 제2 싸이클 직후의 복수의 싸이클의 개수는 동일할 수 있다.

[0054] 그리고, 제어부(120)는 제1 싸이클 직후의 복수의 싸이클 중 최초 싸이클에서, 제1 연산 소자들 각각에 인접한 제1 연산 소자가 제1 엘리먼트를 제공하도록 제1 연산 소자들을 제어하며, 제2 싸이클 직후의 복수의 싸이클 중 최초 싸이클에서, 제2 연산 소자들 각각에 인접한 제2 연산 소자가 제2 엘리먼트를 제공하도록 제2 연산 소자들을 제어할 수 있다.

[0055] 또한, 제어부(120)는 제1 싸이클 직후의 복수의 싸이클 중 최초 싸이클 이후의 싸이클 동안, 제1 연산 소자들 각각에 인접한 제1 연산 소자가 직전 싸이클에서의 연산 데이터를 제공하도록 제1 연산 소자들을 제어하며, 제2 싸이클 직후의 복수의 싸이클 중 최초 싸이클 이후의 싸이클 동안, 제2 연산 소자들 각각에 인접한 제2 연산 소자가 직전 싸이클에서의 연산 데이터를 제공하도록 제2 연산 소자들을 제어할 수 있다.

[0056] 한편, 제어부(120)는 대상 데이터에 포함된 복수의 엘리먼트 중 제3 로우에 포함된 제3 엘리먼트들을 각각 복수의 연산 소자(110) 중 제3 로우에 배열된 제3 연산 소자들로 입력하며, 제3 연산 소자들 각각이, 인접한 제3 연산 소자로부터 제공된 데이터를 입력된 제3 엘리먼트와 연산하도록 제3 연산 소자들을 제어하고, 제3 연산 소자들 각각이, 제2 연산 소자들 중 동일한 컬럼에서 인접한 제2 연산 소자로부터 제공된 연산 데이터를 기저장된 연산 데이터와 연산하도록 제3 연산 소자들을 제어하여 풀링 결과를 획득할 수 있다. 여기서, 인접한 제2 연산 소자로부터 제공된 연산 데이터는 인접한 제2 연산 소자와 동일한 컬럼에서 인접한 제1 연산 소자로부터 제공된 연산 데이터와 인접한 제2 연산 소자의 기저장된 연산 데이터의 연산 결과일 수 있다.

[0057] 한편, 제어부(120)는 제1 연산 소자들 각각이, 인접한 제1 연산 소자로부터 제공된 데이터를 입력된 제1 엘리먼트와 덧셈 연산하도록 제1 연산 소자들을 제어하고, 제2 연산 소자들 각각이, 인접한 제2 연산 소자로부터 제공된 데이터를 입력된 제2 엘리먼트와 덧셈 연산하도록 제2 연산 소자들을 제어하며, 제2 연산 소자들 각각이, 제1 연산 소자들 중 동일한 컬럼에서 인접한 제1 연산 소자로부터 제공된 연산 데이터를 기저장된 연산 데이터와 덧셈 연산하도록 제2 연산 소자들을 제어하여 평균값 풀링 결과를 획득할 수 있다.

[0058] 또는, 제어부(120)는 제1 연산 소자들 각각이, 인접한 제1 연산 소자로부터 제공된 제1 엘리먼트를 입력된 제1 엘리먼트와 비교 연산하도록 제1 연산 소자들을 제어하고, 제2 연산 소자들 각각이, 인접한 제2 연산 소자로부터 제공된 제2 엘리먼트를 입력된 제2 엘리먼트와 비교 연산하도록 제2 연산 소자들을 제어하며, 제2 연산 소자들 각각이, 제1 연산 소자들 중 동일한 컬럼에서 인접한 제1 연산 소자로부터 제공된 제1 엘리먼트를 기저장된 제2 엘리먼트와 비교 연산하도록 제2 연산 소자들을 제어하여 최댓값(max) 풀링 결과를 획득할 수 있다.

[0059] 한편, 복수의 연산 소자(110) 각각은 제1 레지스터 및 제2 레지스터를 포함하고, 제어부(120)는 제1 연산 소자들 각각이, 인접한 제1 연산 소자의 제1 레지스터 또는 제2 레지스터로부터 제공된 데이터를 제1 연산 소자들 각각의 제1 레지스터에 저장된 제1 엘리먼트와 연산하여 제1 연산 소자들 각각의 제2 레지스터에 저장하도록 제1 연산 소자들을 제어하고, 제2 연산 소자들 각각이, 인접한 제2 연산 소자의 제1 레지스터 또는 제2 레지스터로부터 제공된 데이터를 제2 연산 소자들 각각의 제1 레지스터에 저장된 제2 엘리먼트와 연산하여 제2 연산 소자들 각각의 제2 레지스터에 저장하도록 제2 연산 소자들을 제어할 수 있다.

[0060] 여기서, 제어부(120)는 제2 연산 소자들 각각이, 제1 연산 소자들 중 동일한 컬럼에서 인접한 제1 연산 소자의 제1 레지스터로부터 제공된 연산 데이터를 제2 연산 소자들 각각의 제1 레지스터에 저장된 연산 데이터와 연산하여 제2 연산 소자들 각각의 제2 레지스터에 저장하도록 제2 연산 소자들을 제어할 수 있다.

[0061] 이상과 같이 제어부(120)는 복수의 연산 소자(110)를 제어하여 풀링 결과를 획득할 수 있으며, 좀더 구체적인 방법에 대하여는 이후의 도면을 통해 설명하고, 먼저 프로세서(100)의 구조를 설명한다.

[0062] 도 2b는 본 개시의 일 실시 예에 따른 프로세서(100)의 세부 구성을 나타내는 도면이다.

[0063] 복수의 연산 소자(110) 각각은 우측 및 하측으로 데이터를 전송할 수 있다. 또한, 도 2b에 도시하진 않았으나, 복수의 연산 소자(110) 각각은 프로세서(100)의 외부에 구비된 메모리로부터 데이터를 수신할 수 있다. 또는, 복수의 연산 소자(110) 각각은 프로세서(100)의 내부에 구비된 On-chip 메모리(ex, 캐시 메모리, 레지스터)로부터 데이터를 수신할 수도 있다.

[0064] 다만, 이에 한정되는 것은 아니며, 도 2b의 도면은 풀링을 수행하기 위한 최소 동작을 표현하기 위한 도면으로, 복수의 연산 소자(110)는 컨볼루션을 수행하기 위한 데이터 패스(path)를 더 포함할 수도 있다. 예를 들어, 복수의 연산 소자(110) 각각은 좌측 및 상측으로 데이터를 전송할 수도 있다.

[0065] 제어부(120)는 복수의 연산 소자(110)의 로우 단위로 명령어를 입력하여 복수의 연산 소자(110)를 제어할 수 있다. 즉, 제어부(120)는 동일한 로우에 포함된 연산 소자들을 동일한 명령어로 처리하는 SIMD 방법으로 복수의 연산 소자(110)를 제어할 수 있다. 예를 들어, 제어부(120)는 제1 명령어를 입력하여 제1 연산 소자들을 제어하고, 제2 명령어를 입력하여 제2 연산 소자들을 제어할 수 있다. 즉, 제어부(120)는 동일한 로우에 포함된 연산 소자들이 동일한 동작을 수행하도록 제어할 수 있다.

[0066] 또한, 제어부(120)는 대상 데이터에 포함된 복수의 엘리먼트 중 동일한 로우에 포함된 엘리먼트들을 그룹핑하고, 그룹핑된 엘리먼트들 각각을 제1 연산 소자들로 입력할 수 있다. 그리고, 제어부(120)는 제1 명령어로 제1 연산 소자들을 제어하여, 그룹핑된 엘리먼트들에 대하여 동일한 연산을 수행할 수 있다. 즉, 제어부(120)는 SIMD 방법으로 제1 연산 소자들을 제어함에 따라, 그룹핑된 엘리먼트들에 대하여 동일한 연산을 병렬적으로 수행할 수 있다. 이러한 동작을 통해, 제어부(120)는 복수의 로컬 영역 각각에 대한 복수의 풀링 결과를 동시에 획득할 수 있다.

[0067] 여기서, 제어부(120)는 제1 연산 소자들의 개수에 기초하여 그룹핑되는 엘리먼트들의 개수를 결정할 수 있다. 예를 들어, 제어부(120)는 제1 연산 소자들의 개수가 5개이면, 대상 데이터에 포함된 복수의 엘리먼트 중 동일한 로우에 포함된 5개의 엘리먼트들을 그룹핑할 수 있다. 다만, 이에 한정되는 것은 아니며, 제어부(120)는 그룹핑되는 엘리먼트들의 개수를 제1 연산 소자들의 개수 미만으로 결정할 수도 있다.

[0068] 한편, 본 개시에 따른 풀링의 경우, 명령어의 로테이션(rotation)이 가능하다. 예를 들어, 제어부(120)가 제1 싸이클에서 제1 명령어를 입력하여 제1 연산 소자들을 제어한 경우, 제2 싸이클에서는 제1 명령어를 이용하여 제2 연산 소자들을 제어할 수 있다. 이에 대한 구체적인 설명은 후술한다.

[0069] 도 2c는 본 개시의 일 실시 예에 따른 복수의 연산 소자(110) 각각의 세부 구성을 나타내는 도면이다. 복수의 연산 소자(110)는 모두 동일한 형태로 구현될 수 있으며, 도 2d와 같이 연결된 상태일 수 있다.

[0070] 연산 소자는 도 2c에 도시된 바와 같이, 제1 레지스터(Data Register, 210), 제2 레지스터(Accumulation Register, 220), 복수의 멀티플렉서(230-1 ~ 230-3) 및 ADD+Max 유닛(240)을 포함할 수 있다. 여기서, 제1 레지스터(210) 및 제2 레지스터(220) 각각은 데이터를 일시적으로 저장하는 고속의 데이터 저장 매체이고, 복수의 멀티플렉서(230-1 ~ 230-3) 각각은 복수의 데이터를 입력받고, 이중 하나의 데이터를 출력하는 회로일 수 있다. ADD+Max 유닛(240)은 가산기(ADD, 241), 1's comps(242) 및 복수의 멀티플렉서(243-1, 243-2)를 포함하는 회로로서, 풀링의 종류에 따라 ADD+Max 유닛(240) 내의 소자들의 동작이 달라질 수 있으며, 이에 대하여는 후술한다.

[0071] 제1 레지스터(210)는 메모리로부터 입력된 데이터 및 제2 레지스터(220)로부터 입력된 데이터 중 하나를 멀티플렉서(230-1)을 통해 수신할 수 있다.

[0072] 멀티플렉서(230-2)는 제1 레지스터(210)로부터 입력된 데이터 및 제2 레지스터(220)로부터 입력된 데이터 중 하나를 ADD+Max 유닛(240)으로 제공하거나, 우측의 연산 소자 또는 하측의 연산 소자로 제공할 수 있다.

[0073] 멀티플렉서(230-3)는 좌측의 연산 소자로부터 제공된 데이터 및 상측의 연산 소자로부터 제공된 데이터 중 하나를 ADD+Max 유닛(240)으로 제공할 수 있다.

[0074] ADD+Max 유닛(240)은 멀티플렉서(230-2)로부터 제공된 데이터 및 멀티플렉서(230-3)로부터 제공된 데이터를 연산하여 제2 레지스터(220)로 출력할 수 있다. ADD+Max 유닛(240)에 포함된 1's comps(242) 및 복수의 멀티플렉서(243-1, 243-2)는 프로세서(100)가 최댓값 풀링을 수행하는지 또는 평균값 풀링을 수행하는지에 따라 그 동작이 결정될 수 있다.

[0075] 먼저, 최댓값 풀링을 수행하는 경우, 대상 데이터의 두 엘리먼트 중 더 큰 엘리먼트가 식별될 필요가 있다. 이 경우, 두 엘리먼트를 비교하기 위해 두 엘리먼트 간 뺄셈 연산을 수행하고, 연산 결과가 양수인지 음수인지에 따라 더 큰 엘리먼트의 식별이 가능하다. 여기서, 뺄셈 연산은 1's complement 연산 및 덧셈 연산으로 대체될 수 있다.

[0076] 따라서, 최댓값 풀링을 수행하는 경우, 멀티플렉서(243-1)는 1's comps(242)로부터 제공된 데이터를 가산기(241)로 제공하고, 가산기(241)는 멀티플렉서(230-3)로부터 입력된 데이터 및 멀티플렉서(243-1)로부터 입력된 데이터의 덧셈 연산을 수행하여 멀티플렉서(243-2)로 제공할 수 있다. 여기서, 1's comps(242)는 1's complement 연산을 수행하는 회로일 수 있다.

[0077] 멀티플렉서(243-2)는 가산기(241)로부터 입력된 데이터가 양수인지 음수인지에 따라 멀티플렉서(230-3)로부터

입력된 데이터 및 멀티플렉서(243-1)로부터 입력된 데이터 중 하나를 출력할 수 있다.

[0078] 이러한 동작을 통해 두 엘리먼트 간 크기를 비교할 수 있으며, 제어부(120)는 이러한 동작을 로컬 영역 전체에 걸쳐 수행하도록 복수의 연산 소자(110)를 제어하여 최댓값 풀링 결과를 획득할 수 있다.

[0079] 한편, 평균값 풀링을 수행하는 경우, 대상 데이터의 두 엘리먼트가 합산될 필요가 있다. 그에 따라, 멀티플렉서(243-1)는 멀티플렉서(230-2)로부터 입력된 데이터를 가산기(241)로 제공하고, 가산기(241)는 멀티플렉서(230-3)로부터 입력된 데이터 및 멀티플렉서(243-1)로부터 입력된 데이터의 덧셈 연산을 수행하여 멀티플렉서(243-2)로 제공할 수 있다.

[0080] 멀티플렉서(243-2)는 가산기(241)로부터 입력된 데이터를 출력할 수 있다.

[0081] 이러한 동작을 통해 두 엘리먼트를 합산할 수 있으며, 제어부(120)는 이러한 동작을 로컬 영역 전체에 걸쳐 수행하도록 복수의 연산 소자(110)를 제어하고, 로컬 영역 전체에 포함된 엘리먼트들의 개수로 나눗셈 연산을 수행하여 평균값 풀링 결과를 획득할 수 있다.

[0082] 상술한 바와 같이, 연산 소자는 풀링의 종류에 따라 상이하게 동작할 수 있다. 또한, 후술하는 바와 같이, 연산 소자는 싸이클마다 상이하게 동작할 수 있다.

[0083] 제어부(120)는 복수의 멀티플렉서(230-1 ~ 230-3, 243-1, 243-2)를 제어하여, 각 싸이클 별로 연산 소자의 동작을 제어할 수 있다.

[0084] 한편, 도 2c에서는 최댓값 풀링 및 평균값 풀링을 모두 수행할 수 있는 ADD+Max 유닛(240)을 도시하였으나, 이에 한정되는 것은 아니다. 예를 들어, 평균값 풀링만을 수행하는 경우, 연산 소자는 ADD+Max 유닛(240) 대신 가산기만을 포함할 수도 있다.

[0085] 그리고, 도 2a 내지 도 2d에서는 풀링 결과를 획득하기 위한 제어부(120)의 동작을 설명하였으나, 제어부(120)는 복수의 연산 소자(110)를 제어하여 컨볼루션 결과를 획득할 수도 있다.

[0086] 도 3 내지 도 8은 본 개시의 일 실시 예에 따른 복수의 연산 소자(110)의 동작을 설명하기 위한 도면들이다. 도 3 내지 도 8에서는 풀링의 단위가 되는 영역의 크기가 3 × 3이고, 스트라이드 1을 가정한다. 그리고, 설명의 편의를 위해 제어부(120)가 평균값 풀링에 따라 덧셈 연산을 수행하는 것으로 설명한다.

[0087] 먼저, 도 3에 도시된 바와 같이, 대상 데이터는 5 × 6의 매트릭스 형태이고, 복수의 연산 소자(110)는 5 × 6의 매트릭스 형태로 배열된 것으로 가정하였다. 다만, 이는 일 실시 예에 불과하고, 대상 데이터는 얼마든지 다양한 형태일 수 있고, 복수의 연산 소자(110)의 개수 및 배열 역시 다를 수도 있다.

[0088] 도 3에서 1로 기재된 부분은 엘리먼트들 간 덧셈 연산이 수행되지 않은 상태를 나타내고, 풀링을 위한 덧셈 연산이 수행되지 않은 최초의 대상 데이터를 나타낸다. 도 3에서 9로 기재된 부분은 9개의 엘리먼트들의 덧셈 연산이 수행된 상태를 나타내며, 풀링 결과를 나타낸다. 예를 들어, 3 × 3의 제1 로컬 영역(310-1)에 대한 풀링이 수행되어 제1 연산 데이터(310-2)가 획득되고, 3 × 3의 제2 로컬 영역(320-1)에 대한 풀링이 수행되어 제2 연산 데이터(320-2)가 획득될 수 있다. 즉, 도 3의 숫자는 엘리먼트의 사용 개수를 의미한다.

[0089] 도 4는 각 싸이클(Time) 별로 복수의 연산 소자(110)의 동작을 나타낸다. 먼저, 도 4의 첫 번째 라인은 복수의 연산 소자(110)의 첫 번째 로우에 포함된 제1 연산 소자들의 제1 레지스터 및 제2 레지스터에 저장된 데이터를 나타낸다. 즉, 제어부(120)는 Time 1에서 대상 데이터에 포함된 복수의 엘리먼트 중 첫 번째 로우에 포함된 제1 엘리먼트들을 각각 복수의 연산 소자(110) 중 첫 번째 로우에 배열된 제1 연산 소자들의 제1 레지스터로 입력할 수 있다. 도 4의 Time 1에서는 제1 엘리먼트들이 입력된 상태이고, 덧셈 연산이 수행된 것은 아니므로 1로서 표시되었다.

[0090] 제어부(120)는 Time 2에서, 제1 연산 소자들 각각이 우측으로 인접한 제1 연산 소자로 제1 레지스터에 저장된 제1 엘리먼트를 제공하도록 제1 연산 소자들을 제어할 수 있다. 그리고, 제어부(120)는 Time 2에서, 제1 연산 소자들 각각이 좌측으로부터 입력된 제1 엘리먼트 및 Time 1에서 입력된 제1 엘리먼트를 덧셈 연산하고, 연산 결과를 나타내는 연산 데이터를 제2 레지스터에 저장하도록 제1 연산 소자들을 제어할 수 있다. 도 4의 Time 2에서는 제1 연산 소자들 각각의 제1 레지스터는 여전히 Time 1에서 입력된 제1 엘리먼트를 저장하고 있으므로 1로서 표시되었고, 제1 연산 소자들 각각의 제2 레지스터는 두 개의 제1 엘리먼트들 간의 연산 데이터를 저장하고 있으므로 2로서 표시되었다.

[0091] 좌측 4개의 제1 연산 소자들의 구체적인 동작(410)을 도 5에 도시하였다. Time 2에서 제1 연산 소자들은 각각

제1 레지스터에 저장된 제1 엘리먼트를 우측의 제1 연산 소자의 ADD+Max 유닛으로 제공할 수 있다. 그리고, 제1 연산 소자들은 각각 제1 레지스터에 저장된 제1 엘리먼트를 ADD+Max 유닛으로 입력할 수 있다. 제1 연산 소자들 각각의 ADD+Max 유닛은 복수의 제1 엘리먼트들을 덧셈 연산하여 제2 레지스터에 저장할 수 있다.

[0092] 다시 도 4의 Time 3을 설명하면, 제어부(120)는 제1 연산 소자들 각각이 우측으로 인접한 제1 연산 소자로 제2 레지스터에 저장된 연산 데이터를 제공하도록 제1 연산 소자들을 제어할 수 있다. 그리고, 제어부(120)는 Time 3에서, 제1 연산 소자들 각각이 좌측으로부터 입력된 연산 데이터 및 Time 1에서 입력된 제1 엘리먼트를 덧셈 연산하고, 연산 결과를 나타내는 연산 데이터를 제2 레지스터에 저장하며, 제1 레지스터를 삭제하도록 제1 연산 소자들을 제어할 수 있다. Time 3에서 제2 레지스터에 저장된 연산 데이터는 새로운 연산 데이터로 업데이트될 수 있다. 도 4의 Time 3에서는 제1 연산 소자들 각각의 제2 레지스터는 세 개의 제1 엘리먼트들 간의 연산 데이터를 저장하고 있으므로 3으로서 표시되었다.

[0093] 또한, 제어부(120)는 Time 3에서 대상 데이터에 포함된 복수의 엘리먼트 중 두 번째 로우에 포함된 제2 엘리먼트들을 각각 복수의 연산 소자(110) 중 두 번째 로우에 배열된 제2 연산 소자들의 제1 레지스터로 입력할 수 있다.

[0094] 제어부(120)는 Time 4에서 제1 연산 소자들 각각의 제2 레지스터에 저장된 연산 데이터를 제1 레지스터로 이동하도록 제1 연산 소자들을 제어하며, Time 6까지 제1 연산 소자들을 추가 제어하지 않는다.

[0095] 제어부(120)는 Time 2부터 Time 4까지의 제1 연산 소자들에 대한 제어 방법으로 Time 4부터 Time 6까지 제2 연산 소자들을 동일하게 제어할 수 있다.

[0096] 제어부(120)는 Time 5에서 대상 데이터에 포함된 복수의 엘리먼트 중 세 번째 로우에 포함된 제3 엘리먼트들을 각각 복수의 연산 소자(110) 중 세 번째 로우에 배열된 제3 연산 소자들의 제1 레지스터로 입력할 수 있다.

[0097] 제어부(120)는 Time 2부터 Time 3까지의 제1 연산 소자들에 대한 제어 방법으로 Time 6부터 Time 7까지 제3 연산 소자들을 동일하게 제어할 수 있다.

[0098] 제어부(120)는 Time 7에서, 제1 연산 소자들 각각이 하측으로 인접한 제2 연산 소자로 제1 레지스터에 저장된 연산 데이터를 제공하도록 제1 연산 소자들을 제어할 수 있다. 그리고, 제어부(120)는 Time 7에서, 제2 연산 소자들 각각이 제1 연산 소자로부터 입력된 연산 데이터 및 제2 연산 소자들 각각의 제1 레지스터에 저장된 연산 데이터를 덧셈 연산하고, 연산 결과를 나타내는 연산 데이터를 제2 레지스터에 저장하도록 제2 연산 소자들을 제어할 수 있다. 도 4의 Time 7에서는 제2 연산 소자들 각각의 제1 레지스터는 여전히 Time 6에서 입력된 연산 데이터를 저장하고 있으므로 3으로서 표시되었고, 제2 연산 소자들 각각의 제2 레지스터는 여섯 개의 제1 엘리먼트들 간의 연산 데이터를 저장하고 있으므로 6으로서 표시되었다.

[0099] 2개의 제1 연산 소자들 및 2개의 제2 연산 소자들의 구체적인 동작(420)을 도 6에 도시하였다. Time 7에서 제1 연산 소자들은 각각 제1 레지스터에 저장된 연산 데이터를 하측의 제2 연산 소자의 ADD+Max 유닛으로 제공할 수 있다. 그리고, 제2 연산 소자들은 각각 제1 레지스터에 저장된 연산 데이터를 ADD+Max 유닛으로 입력할 수 있다. 제2 연산 소자들 각각의 ADD+Max 유닛은 제1 연산 소자들로부터 제공된 연산 데이터 및 제2 연산 소자들 각각의 제1 레지스터로부터 제공된 연산 데이터를 덧셈 연산하여 제2 레지스터에 저장할 수 있다.

[0100] 그리고, 제어부(120)는 Time 7에서 대상 데이터에 포함된 복수의 엘리먼트 중 네 번째 로우에 포함된 제4 엘리먼트들을 각각 복수의 연산 소자(110) 중 네 번째 로우에 배열된 제4 연산 소자들의 제1 레지스터로 입력할 수 있다.

[0101] 제어부(120)는 Time 8에서, 제2 연산 소자들 각각이 하측으로 인접한 제3 연산 소자로 제2 레지스터에 저장된 연산 데이터를 제공하도록 제2 연산 소자들을 제어할 수 있다. 그리고, 제어부(120)는 Time 8에서, 제3 연산 소자들 각각이 제2 연산 소자로부터 입력된 연산 데이터 및 제3 연산 소자들 각각의 제2 레지스터에 저장된 연산 데이터를 덧셈 연산하고, 연산 결과를 나타내는 연산 데이터를 제2 레지스터에 저장하도록 제2 연산 소자들을 제어할 수 있다. 그리고, 제어부(120)는 Time 8에서 제3 연산 소자들 각각의 제2 레지스터에 저장된 연산 데이터를 제1 레지스터로 이동하도록 제3 연산 소자들을 제어할 수 있다. 도 4의 Time 8에서는 제3 연산 소자들 각각의 제1 레지스터는 Time 8에서 입력된 연산 데이터를 저장하고 있으므로 3으로서 표시되었고, 제3 연산 소자들 각각의 제2 레지스터는 아홉 개의 제1 엘리먼트들 간의 연산 데이터를 저장하고 있으므로 9로서 표시되었다.

[0102] 2개의 제2 연산 소자들 및 2개의 제3 연산 소자들의 구체적인 동작(430)을 도 7에 도시하였다. Time 8에서 제2 연산 소자들은 각각 제2 레지스터에 저장된 연산 데이터를 하측의 제3 연산 소자의 ADD+Max 유닛으로 제공할 수

있다. 그리고, 제3 연산 소자들은 각각 제2 레지스터에 저장된 연산 데이터를 ADD+Max 유닛으로 입력할 수 있다. 제3 연산 소자들 각각의 ADD+Max 유닛은 제2 연산 소자들로부터 제공된 연산 데이터 및 제3 연산 소자들 각각의 제2 레지스터로부터 제공된 연산 데이터를 덧셈 연산하여 제2 레지스터에 저장할 수 있다.

[0103] 제어부(120)는 제3 연산 소자들의 제2 레지스터에 저장된 연산 데이터를 풀링 결과로서 독출할 수 있다. 도 4의 Time 8에 따르면, 풀링 결과는 4개이며, 이는 4개의 로컬 영역에 대한 풀링 결과로서, 병렬 연산이 수행되었음을 의미한다. 즉, 복수의 연산 소자(110)의 열의 개수가 증가할수록 병렬 처리 능력이 향상될 수 있다.

[0104] 제어부(120)는 Time 7부터 Time 8까지의 제2 연산 소자들에 대한 제어 방법으로 Time 9부터 Time 10까지 제3 연산 소자들을 제어하고, Time 7부터 Time 8까지의 제3 연산 소자들에 대한 제어 방법으로 Time 9부터 Time 10까지 제4 연산 소자들을 제어할 수 있다.

[0105] 즉, 제어부(120)는 이상과 같은 제어 방법을 반복 적용하여 풀링 결과를 획득할 수 있다. 특히, 로우 방향의 엘리먼트들 간의 연산 결과를 연산 소자가 저장하고, 이를 하측의 연산 소자로 제공함에 따라 종래의 중복 연산 문제를 해결할 수 있다.

[0106] 또한, 종래 기술과는 달리 복수의 연산 소자(110)는 대상 데이터에 포함된 복수의 엘리먼트를 각각 한 번만 입력받으므로, 데이터의 중복 입력 문제를 해결할 수 있다.

[0107] 한편, 제어부(120)의 싸이클 별 동작 결과를 도 8에 간략히 도시하였다. 설명의 편의를 위해 복수의 연산 소자(110)가 복수의 열의 연산 소자를 포함함에 따른 병렬 처리를 생략한 형태로 도시하였다. 도 8의 가로축은 Time을 나타내고, 세로축은 복수의 연산 소자(110)의 로우를 나타낸다. 도 8에서 F는 대상 데이터의 일부가 복수의 연산 소자(110)로 입력되는 시점을 나타내고, 숫자는 엘리먼트의 사용 개수를 의미한다.

[0108] 즉, Time 8에서 최초로 풀링 결과가 획득되고, 이후 두 싸이클 간격으로 추가적인 풀링 결과가 획득되며, 이는 로우 방향의 엘리먼트들 간의 연산 결과를 연산 소자가 저장하고 있기 때문이다.

[0109] 또한, 두 싸이클 간격으로 대상 데이터의 일부가 입력되어, 메모리 대역폭을 균일하게 유지하며 피크 대역폭(Peak bandwidth)을 낮출 수 있다.

[0110] 도 9a 내지 도 11c는 본 개시의 다양한 실시 예에 따른 싸이클 별 제어부(120)의 동작 방법을 나타내는 도면들이다.

[0111] 도 3 내지 도 8에서는 풀링의 단위가 되는 영역의 크기가 3 × 3이고, 스트라이드 1을 가정하였으나, 도 9a 내지 도 9e와 같이 풀링의 단위가 되는 영역의 크기가 4 × 4이거나, 도 10a 내지 도 10d와 같이 풀링의 단위가 되는 영역의 크기가 3 × 3이거나, 도 11a 내지 도 11c와 같이 풀링의 단위가 되는 영역의 크기가 2 × 2일 수도 있다.

[0112] 또한, 풀링의 단위가 되는 영역의 크기 중 일측이 스트라이드와 동일하면, 로우 방향의 엘리먼트들 간의 연산 결과가 주기적으로 이용될 필요가 없기 때문에 도 9e, 도 10d, 도 11c와 같은 동작도 가능하다.

[0113] 도 9a 내지 도 11c와 같은 동작 방법은 프로세서(100)의 외부에 구비된 메모리에 저장된 상태일 수 있다. 제어부(120)는 풀링 명령이 입력되면, 풀링 조건(ex, 풀링의 단위가 되는 영역의 크기 및 스트라이드)에 대응되는 동작 방법을 메모리로부터 독출하여 풀링을 수행할 수 있다.

[0114] 다만, 이에 한정되는 것은 아니며, 프로세서(100)는 도 9a 내지 도 11c와 같은 동작 방법이 저장된 내부 메모리(ex, 캐시 메모리, 레지스터)를 더 포함하며, 제어부(120)는 풀링 명령이 입력되면, 풀링 조건에 대응되는 동작 방법을 내부 메모리로부터 독출하여 풀링을 수행할 수도 있다.

[0115] 도 12는 본 개시의 일 실시 예에 따른 복수의 연산 소자(110)의 Utilization을 높이는 방법을 설명하기 위한 도면이다.

[0116] 도 12의 상측 도면은 풀링의 단위가 되는 영역의 크기가 3 × 3이고, 스트라이드 2인 경우의 제어부(120)의 동작 방법을 나타낸다. 이 경우, 첫 번째 로우에 포함된 연산 소자는 Time 5에서 연산 데이터를 두 번째 로우에 제공한 이후, 추가 동작을 수행하지 않는다.

[0117] 따라서, 도 12의 하측 도면과 같이, 제어부(120)는 Time 5에서 대상 데이터에 포함된 추가 엘리먼트를 첫 번째 로우에 포함된 연산 소자로 입력하여 풀링을 수행할 수 있다. 연산 소자는 제1 레지스터 및 제2 레지스터를 포함하므로, Time 5에서 추가 엘리먼트가 입력되더라도 문제가 발생하지 않는다.

[0118] 도 13은 본 개시의 일 실시 예에 따른 명령어의 로테이션(rotation)을 설명하기 위한 도면이다.

[0119] 도 13에 도시된 바와 같이, 제어부(120)는 Time 13에서 첫 번째 로우에 포함된 제1 연산 소자들부터 열 세번째 로우에 포함된 제13 연산 소자들에 명령어들을 입력할 수 있다. 이때, 입력되는 명령어는 로우 단위로 상이하므로 총 13개의 명령어가 입력될 수 있다.

[0120] 그리고, 제어부(120)는 Time 13에서 입력한 명령어들을 Time 14에서 두 번째 로우에 포함된 제2 연산 소자들부터 열 네번째 로우에 포함된 제14 연산 소자들에 입력할 수 있다.

[0121] 즉, 제어부(120)는 복수의 명령어를 로테이션하여 복수의 연산 소자(110)를 제어할 수 있다. 가령, 하나의 명령어는 제1 로우에 포함된 제1 연산 소자들에 입력된 후, 순차적으로 아래의 로우에 포함된 연산 소자들에 입력되며, 마지막 로우에 포함된 연산 소자들에 입력된 후, 다시 제1 연산 소자들에 입력될 수 있다. 그에 따라, 제어부(120)는 도 2b에 도시된 바와 같이, 복수의 명령어를 복수의 연산 소자(110)의 로우 단위로 입력하는 형태로 구현될 수 있다.

[0122] 도 14는 본 개시의 일 실시 예에 따른 효과를 설명하기 위한 도면이다.

[0123] 상술한 바와 같이, 프로세서(100)는 풀링을 수행할 수 있다. 또한, 복수의 연산 소자(110) 각각은 컨볼루션을 수행하기 위한 구성을 더 포함하기 때문에, 프로세서(100)는 컨볼루션을 수행할 수도 있다.

[0124] 즉, 복수의 연산 소자(110)는 도 14에 도시된 바와 같이, 풀링을 수행하거나 컨볼루션을 수행하는 상태가 유지되며, 그에 따라 Load-balancing 문제가 해결될 수 있다.

[0125] 도 15는 본 개시의 일 실시 예에 따른 프로세서(100)를 이용하는 전자 장치(1000)를 설명하기 위한 도면이다. 도 15에 도시된 바와 같이, 전자 장치(1000)는 프로세서(100) 및 메모리(200)를 포함할 수 있다.

[0126] 전자 장치(1000)는 인공지능 알고리즘을 학습하거나 인공지능 모델에 따른 동작을 수행하는 장치로서, 컴퓨터, 노트북, 서버, 태블릿, 스마트폰 등과 같은 장치일 수 있다. 다만, 이에 한정되는 것은 아니며, 전자 장치(1000)는 인공지능 알고리즘을 학습하거나 인공지능 모델에 따른 동작을 수행할 수 있다면 어떠한 장치라도 무방하다.

[0127] 전자 장치(1000)가 인공지능 알고리즘을 학습하거나 인공지능 모델에 따른 동작을 수행하는 과정에서, 프로세서(100)는 컨볼루션 또는 풀링을 수행할 수 있다. 특히, 프로세서(100)는 2차원 계산기 구조(2D Array)의 복수의 연산 소자(110)를 이용하여 컨볼루션 또는 풀링을 수행할 수 있다. 예를 들어, 프로세서(100)는 제1 시간 구간 동안 복수의 연산 소자(110)를 이용하여 컨볼루션을 수행하고, 제1 시간 구간 이후의 제2 시간 구간 동안 복수의 연산 소자(110)를 이용하여 풀링을 수행할 수 있다.

[0128] 전자 장치(1000)가 인공지능 알고리즘을 학습하는 경우, 프로세서(100)는 메모리(200)에 저장된 인공지능 알고리즘에 따라 컨볼루션 또는 풀링을 수행하고, 학습 과정에 있는 인공지능 모델을 업데이트할 수 있다.

[0129] 전자 장치(1000)가 인공지능 모델에 따른 동작을 수행하는 경우, 프로세서(100)는 메모리(200)에 저장된 인공지능 모델을 독출하고, 입력 데이터를 인공지능 모델에 적용하여 출력 데이터를 획득할 수 있다. 프로세서(100)는 입력 데이터를 인공지능 모델에 적용하는 과정에서 컨볼루션 또는 풀링을 수행할 수 있다.

[0130] 도 16은 본 개시의 일 실시 예에 따른 프로세서의 제어 방법을 설명하기 위한 흐름도이다.

[0131] 매트릭스 형태로 배열된 복수의 연산 소자(Processing Element)를 포함하며, 복수의 싸이클 동안 상기 복수의 연산 소자를 이용하여 대상 데이터에 대한 풀링(pooling)을 수행하는 프로세서의 제어 방법은, 먼저 대상 데이터에 포함된 복수의 엘리먼트 중 제1 로우(row)에 포함된 제1 엘리먼트들을 각각 복수의 연산 소자 중 제1 로우에 배열된 제1 연산 소자들로 입력한다(S1610). 그리고, 제1 연산 소자들 각각이, 인접한 제1 연산 소자로부터 제공된 데이터를 입력된 제1 엘리먼트와 연산하도록 제1 연산 소자들을 제어하고, 복수의 엘리먼트 중 제2 로우에 포함된 제2 엘리먼트들을 각각 복수의 연산 소자 중 제2 로우에 배열된 제2 연산 소자들로 입력한다(S1620). 그리고, 제2 연산 소자들 각각이, 인접한 제2 연산 소자로부터 제공된 데이터를 입력된 제2 엘리먼트와 연산하도록 제2 연산 소자들을 제어한다(S1630). 그리고, 제2 연산 소자들 각각이, 제1 연산 소자들 중 동일한 컬럼(column)에서 인접한 제1 연산 소자로부터 제공된 연산 데이터를 기저장된 연산 데이터와 연산하도록 제2 연산 소자들을 제어하여 풀링을 수행한다(S1640).

[0132] 여기서, 제1 연산 소자들로 입력하는 단계(S1610)는 제1 싸이클에서, 제1 엘리먼트들을 각각 제1 연산 소자들로 입력하며, 제2 연산 소자들로 입력하는 단계(S1620)는 제1 싸이클 직후의 제2 싸이클에서, 제1 연산 소자들 각

각이, 인접한 제1 연산 소자로부터 제공된 데이터를 입력된 제1 엘리먼트와 연산하도록 제1 연산 소자들을 제어하고, 제2 엘리먼트들을 각각 제2 연산 소자들로 입력하며, 제2 연산 소자들을 제어하는 단계(S1630)는 제2 싸이클 직후의 제3 싸이클에서, 제2 연산 소자들 각각이, 인접한 제2 연산 소자로부터 제공된 데이터를 입력된 제2 엘리먼트와 연산하도록 제2 연산 소자들을 제어하고, 풀링을 수행하는 단계(S1640)는 제3 싸이클 직후의 제4 싸이클에서, 제2 연산 소자들 각각이, 제1 연산 소자들 중 동일한 컬럼에서 인접한 제1 연산 소자로부터 제공된 연산 데이터를 기저장된 연산 데이터와 연산하도록 제2 연산 소자들을 제어하여 풀링을 수행할 수 있다.

[0133] 또는, 제1 연산 소자들로 입력하는 단계(S1610)는 제1 싸이클에서, 제1 엘리먼트들을 각각 제1 연산 소자들로 입력하며, 제2 연산 소자들로 입력하는 단계(S1620)는 제1 싸이클 직후의 복수의 싸이클 동안, 제1 연산 소자들 각각이, 인접한 제1 연산 소자로부터 제공된 데이터를 입력된 제1 엘리먼트와 연산하도록 제1 연산 소자들을 제어하고, 복수의 싸이클 중 하나인 제2 싸이클에서, 제2 엘리먼트들을 각각 제2 연산 소자들로 입력하며, 제2 연산 소자들을 제어하는 단계(S1630)는 제2 싸이클 직후의 복수의 싸이클 동안, 제2 연산 소자들 각각이, 인접한 제2 연산 소자로부터 제공된 데이터를 입력된 제2 엘리먼트와 연산하도록 제2 연산 소자들을 제어하고, 풀링을 수행하는 단계(S1640)는 제2 싸이클 직후의 복수의 싸이클로부터 기설정된 간격의 제3 싸이클에서, 제2 연산 소자들 각각이, 제1 연산 소자들 중 동일한 컬럼에서 인접한 제1 연산 소자로부터 제공된 연산 데이터를 기저장된 연산 데이터와 연산하도록 제2 연산 소자들을 제어하여 풀링을 수행할 수 있다. 여기서, 제1 싸이클 직후의 복수의 싸이클의 개수와 제2 싸이클 직후의 복수의 싸이클의 개수는 동일할 수 있다.

[0134] 또한, 제1 연산 소자들을 제어하는 단계는 제1 싸이클 직후의 복수의 싸이클 중 최초 싸이클에서, 제1 연산 소자들 각각에 인접한 제1 연산 소자가 제1 엘리먼트를 제공하도록 제1 연산 소자들을 제어하며, 제2 연산 소자들을 제어하는 단계(S1630)는 제2 싸이클 직후의 복수의 싸이클 중 최초 싸이클에서, 제2 연산 소자들 각각에 인접한 제2 연산 소자가 제2 엘리먼트를 제공하도록 제2 연산 소자들을 제어할 수 있다.

[0135] 여기서, 제1 연산 소자들을 제어하는 단계는 제1 싸이클 직후의 복수의 싸이클 중 최초 싸이클 이후의 싸이클 동안, 제1 연산 소자들 각각에 인접한 제1 연산 소자가 직전 싸이클에서의 연산 데이터를 제공하도록 제1 연산 소자들을 제어하며, 제2 연산 소자들을 제어하는 단계(S1630)는 제2 싸이클 직후의 복수의 싸이클 중 최초 싸이클 이후의 싸이클 동안, 제2 연산 소자들 각각에 인접한 제2 연산 소자가 직전 싸이클에서의 연산 데이터를 제공하도록 제2 연산 소자들을 제어할 수 있다.

[0136] 한편, 대상 데이터에 포함된 복수의 엘리먼트 중 제3 로우에 포함된 제3 엘리먼트들을 각각 복수의 연산 소자 중 제3 로우에 배열된 제3 연산 소자들로 입력하는 단계 및 제3 연산 소자들 각각이, 인접한 제3 연산 소자로부터 제공된 데이터를 입력된 제3 엘리먼트와 연산하도록 제3 연산 소자들을 제어하는 단계를 더 포함하며, 풀링을 수행하는 단계(S1640)는 제3 연산 소자들 각각이, 제2 연산 소자들 중 동일한 컬럼에서 인접한 제2 연산 소자로부터 제공된 연산 데이터를 기저장된 연산 데이터와 연산하도록 제3 연산 소자들을 제어하여 풀링을 수행할 수 있다. 여기서, 인접한 제2 연산 소자로부터 제공된 연산 데이터는 인접한 제2 연산 소자와 동일한 컬럼에서 인접한 제1 연산 소자로부터 제공된 연산 데이터와 인접한 제2 연산 소자의 기저장된 연산 데이터의 연산 결과일 수 있다.

[0137] 그리고, 제1 연산 소자들을 제어하는 단계는 제1 연산 소자들 각각이, 인접한 제1 연산 소자로부터 제공된 데이터를 입력된 제1 엘리먼트와 덧셈 연산하도록 제1 연산 소자들을 제어하고, 제2 연산 소자들을 제어하는 단계(S1630)는 제2 연산 소자들 각각이, 인접한 제2 연산 소자로부터 제공된 데이터를 입력된 제2 엘리먼트와 덧셈 연산하도록 제2 연산 소자들을 제어하며, 풀링을 수행하는 단계(S1640)는 제2 연산 소자들 각각이, 제1 연산 소자들 중 동일한 컬럼에서 인접한 제1 연산 소자로부터 제공된 연산 데이터를 기저장된 연산 데이터와 덧셈 연산하도록 제2 연산 소자들을 제어하여 평균값(average) 풀링을 수행할 수 있다.

[0138] 또는, 제1 연산 소자들을 제어하는 단계는 제1 연산 소자들 각각이, 인접한 제1 연산 소자로부터 제공된 제1 엘리먼트를 입력된 제1 엘리먼트와 비교 연산하도록 제1 연산 소자들을 제어하고, 제2 연산 소자들을 제어하는 단계(S1630)는 제2 연산 소자들 각각이, 인접한 제2 연산 소자로부터 제공된 제2 엘리먼트를 입력된 제2 엘리먼트와 비교 연산하도록 제2 연산 소자들을 제어하며, 풀링을 수행하는 단계(S1640)는 제2 연산 소자들 각각이, 제1 연산 소자들 중 동일한 컬럼에서 인접한 제1 연산 소자로부터 제공된 제1 엘리먼트를 기저장된 제2 엘리먼트와 비교 연산하도록 제2 연산 소자들을 제어하여 최댓값(max) 풀링을 수행할 수 있다.

[0139] 한편, 복수의 연산 소자 각각은 제1 레지스터 및 제2 레지스터를 포함하고, 제1 연산 소자들을 제어하는 단계는 제1 연산 소자들 각각이, 인접한 제1 연산 소자의 제1 레지스터 또는 제2 레지스터로부터 제공된 데이터를 제1 연산 소자들 각각의 제1 레지스터에 저장된 제1 엘리먼트와 연산하여 제1 연산 소자들 각각의 제2 레지스터에

저장하도록 제1 연산 소자들을 제어하고, 제2 연산 소자들을 제어하는 단계(S1630)는 제2 연산 소자들 각각이, 인접한 제2 연산 소자의 제1 레지스터 또는 제2 레지스터로부터 제공된 데이터를 제2 연산 소자들 각각의 제1 레지스터에 저장된 제2 엘리먼트와 연산하여 제2 연산 소자들 각각의 제2 레지스터에 저장하도록 제2 연산 소자들을 제어할 수 있다.

[0140] 여기서, 풀링을 수행하는 단계(S1640)는 제2 연산 소자들 각각이, 제1 연산 소자들 중 동일한 컬럼에서 인접한 제1 연산 소자의 제1 레지스터로부터 제공된 연산 데이터를 제2 연산 소자들 각각의 제1 레지스터에 저장된 연산 데이터와 연산하여 제2 연산 소자들 각각의 제2 레지스터에 저장하도록 제2 연산 소자들을 제어할 수 있다.

[0141] 이상과 같은 본 발명의 다양한 실시 예에 따르면, 프로세서는 2D Array 구조의 복수의 연산 소자(Processing Element)를 이용하여 컨볼루션 뿐만 아니라 풀링을 수행하게 되어, 하드웨어 이원화 문제, Load-balancing 문제 및 중복 연산 문제를 해결하고, 피크 대역폭(Peak bandwidth)을 낮출 수 있다.

[0142] 한편, 본 개시의 일시 예에 따르면, 이상에서 설명된 다양한 실시 예들은 기기(machine)(예: 컴퓨터)로 읽을 수 있는 저장 매체(machine-readable storage media)에 저장된 명령어를 포함하는 소프트웨어로 구현될 수 있다. 기기는, 저장 매체로부터 저장된 명령어를 호출하고, 호출된 명령어에 따라 동작이 가능한 장치로서, 개시된 실시 예들에 따른 전자 장치(예: 전자 장치(A))를 포함할 수 있다. 명령이 프로세서에 의해 실행될 경우, 프로세서가 직접, 또는 프로세서의 제어 하에 다른 구성요소들을 이용하여 명령에 해당하는 기능을 수행할 수 있다. 명령은 컴파일러 또는 인터프리터에 의해 생성 또는 실행되는 코드를 포함할 수 있다. 기기로 읽을 수 있는 저장매체는, 비일시적(non-transitory) 저장매체의 형태로 제공될 수 있다. 여기서, '비일시적'은 저장매체가 신호(signal)를 포함하지 않으며 실재(tangible)한다는 것을 의미할 뿐 데이터가 저장매체에 반영구적 또는 임시적으로 저장됨을 구분하지 않는다.

[0143] 또한, 본 개시의 일 실시 예에 따르면, 이상에서 설명된 다양한 실시 예들에 따른 방법은 컴퓨터 프로그램 제품(computer program product)에 포함되어 제공될 수 있다. 컴퓨터 프로그램 제품은 상품으로서 판매자 및 구매자 간에 거래될 수 있다. 컴퓨터 프로그램 제품은 기기로 읽을 수 있는 저장 매체(예: compact disc read only memory (CD-ROM))의 형태로, 또는 어플리케이션 스토어(예: 플레이 스토어TM)를 통해 온라인으로 배포될 수 있다. 온라인 배포의 경우에, 컴퓨터 프로그램 제품의 적어도 일부는 제조사의 서버, 어플리케이션 스토어의 서버, 또는 중계 서버의 메모리와 같은 저장 매체에 적어도 일시 저장되거나, 임시적으로 생성될 수 있다.

[0144] 또한, 본 개시의 일 실시 예에 따르면, 이상에서 설명된 다양한 실시 예들은 소프트웨어(software), 하드웨어(hardware) 또는 이들의 조합을 이용하여 컴퓨터(computer) 또는 이와 유사한 장치로 읽을 수 있는 기록 매체 내에서 구현될 수 있다. 일부 경우에 있어 본 명세서에서 설명되는 실시 예들이 프로세서 자체로 구현될 수 있다. 소프트웨어적인 구현에 의하면, 본 명세서에서 설명되는 절차 및 기능과 같은 실시 예들은 별도의 소프트웨어 모듈들로 구현될 수 있다. 소프트웨어 모듈들 각각은 본 명세서에서 설명되는 하나 이상의 기능 및 동작을 수행할 수 있다.

[0145] 한편, 상술한 다양한 실시 예들에 따른 기기의 프로세싱 동작을 수행하기 위한 컴퓨터 명령어(computer instructions)는 비일시적 컴퓨터 판독 가능 매체(non-transitory computer-readable medium)에 저장될 수 있다. 이러한 비일시적 컴퓨터 판독 가능 매체에 저장된 컴퓨터 명령어는 특정 기기의 프로세서에 의해 실행되었을 때 상술한 다양한 실시 예에 따른 기기에서의 처리 동작을 특정 기기가 수행하도록 한다. 비일시적 컴퓨터 판독 가능 매체란 레지스터, 캐쉬, 메모리 등과 같이 짧은 순간 동안 데이터를 저장하는 매체가 아니라 반영구적으로 데이터를 저장하며, 기기에 의해 판독(reading)이 가능한 매체를 의미한다. 비일시적 컴퓨터 판독 가능 매체의 구체적인 예로는, CD, DVD, 하드 디스크, 블루레이 디스크, USB, 메모리카드, ROM 등이 있을 수 있다.

[0146] 또한, 상술한 다양한 실시 예들에 따른 구성 요소(예: 모듈 또는 프로그램) 각각은 단수 또는 복수의 개체로 구성될 수 있으며, 전술한 해당 서브 구성 요소들 중 일부 서브 구성 요소가 생략되거나, 또는 다른 서브 구성 요소가 다양한 실시 예에 더 포함될 수 있다. 대체적으로 또는 추가적으로, 일부 구성 요소들(예: 모듈 또는 프로그램)은 하나의 개체로 통합되어, 통합되기 이전의 각각의 해당 구성 요소에 의해 수행되는 기능을 동일 또는 유사하게 수행할 수 있다. 다양한 실시 예들에 따른, 모듈, 프로그램 또는 다른 구성 요소에 의해 수행되는 동작들은 순차적, 병렬적, 반복적 또는 휴리스틱하게 실행되거나, 적어도 일부 동작이 다른 순서로 실행되거나, 생략되거나, 또는 다른 동작이 추가될 수 있다.

[0147] 이상에서는 본 개시의 바람직한 실시 예에 대하여 도시하고 설명하였지만, 본 개시는 상술한 특정의 실시 예에 한정되지 아니하며, 청구범위에서 청구하는 본 개시의 요지를 벗어남이 없이 당해 개시에 속하는 기술분야에서

통상의 지식을 가진 자에 의해 다양한 변형실시가 가능한 것은 물론이고, 이러한 변형실시들은 본 개시의 기술적 사상이나 전망으로부터 개별적으로 이해되어져서는 안될 것이다.

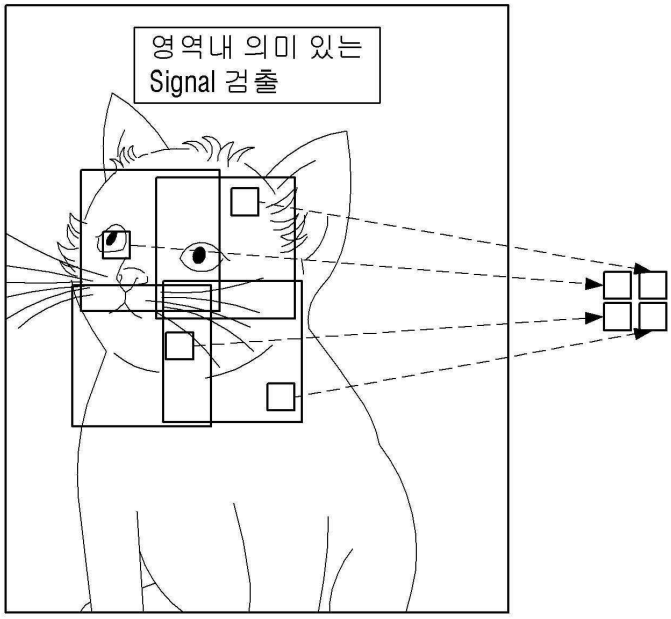
부호의 설명

[0148] 100 : 프로세서 110 : 복수의 연산 소자

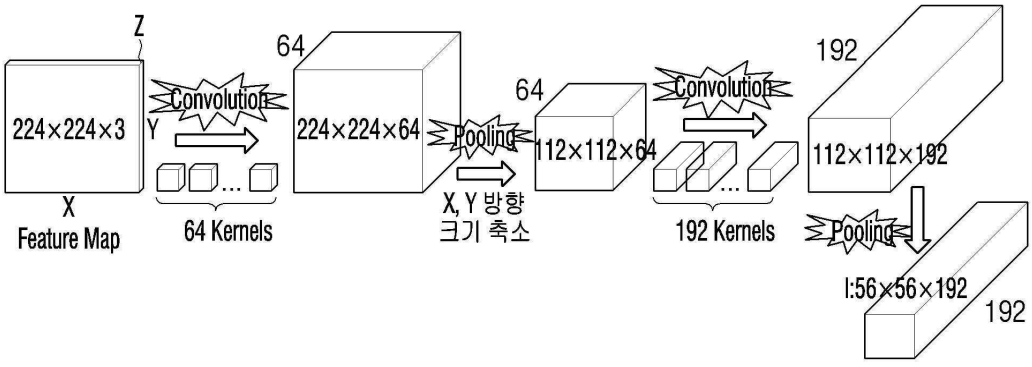
120 : 제어부

도면

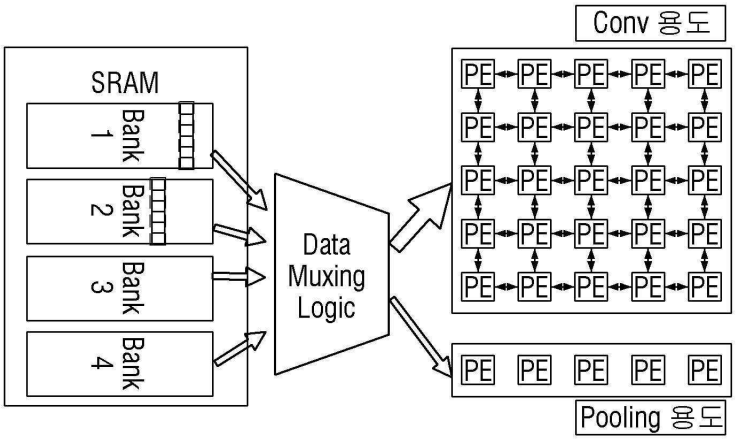
도면1a



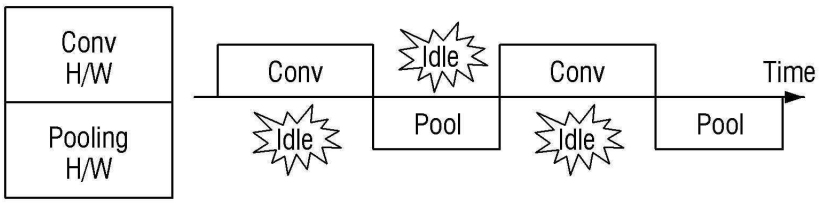
도면1b



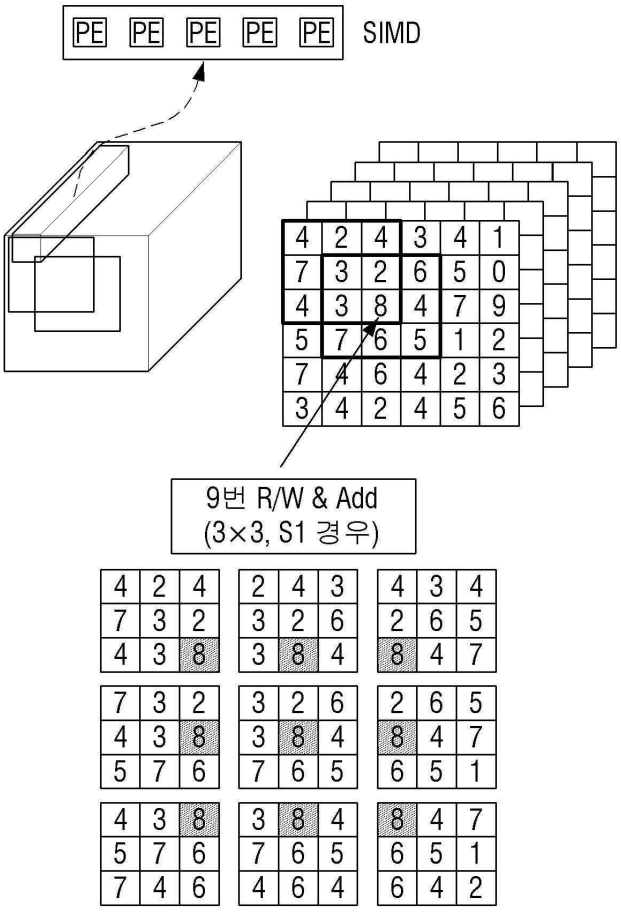
도면1c



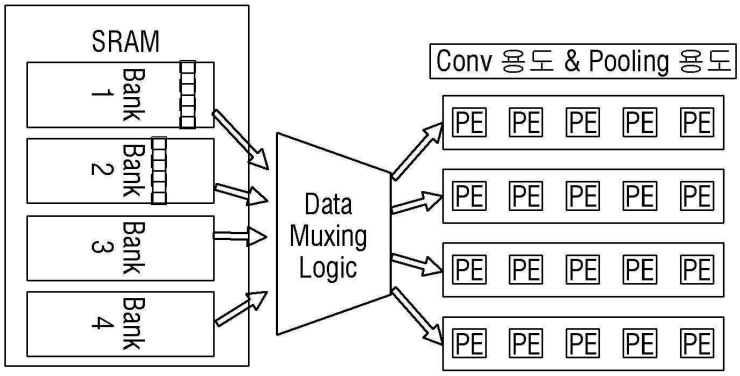
도면1d



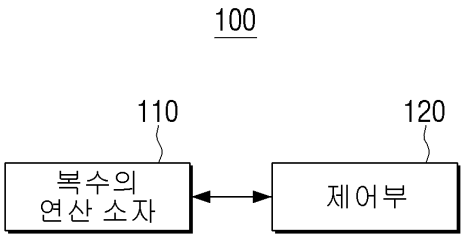
도면1e



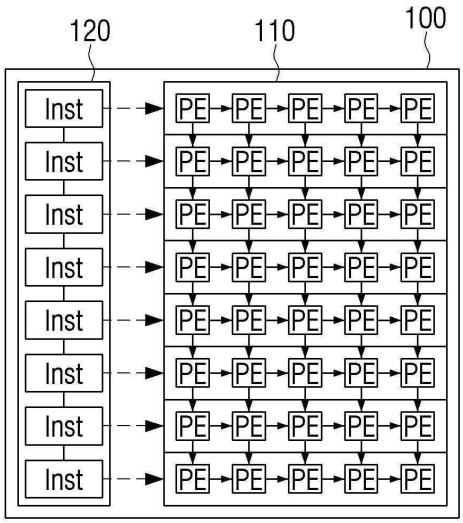
도면1f



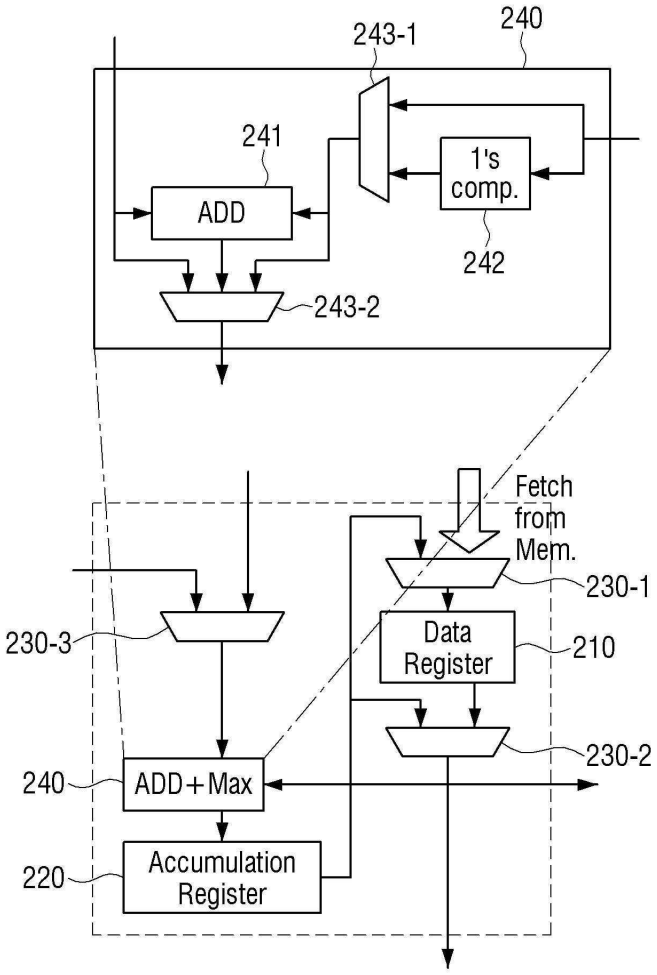
도면2a



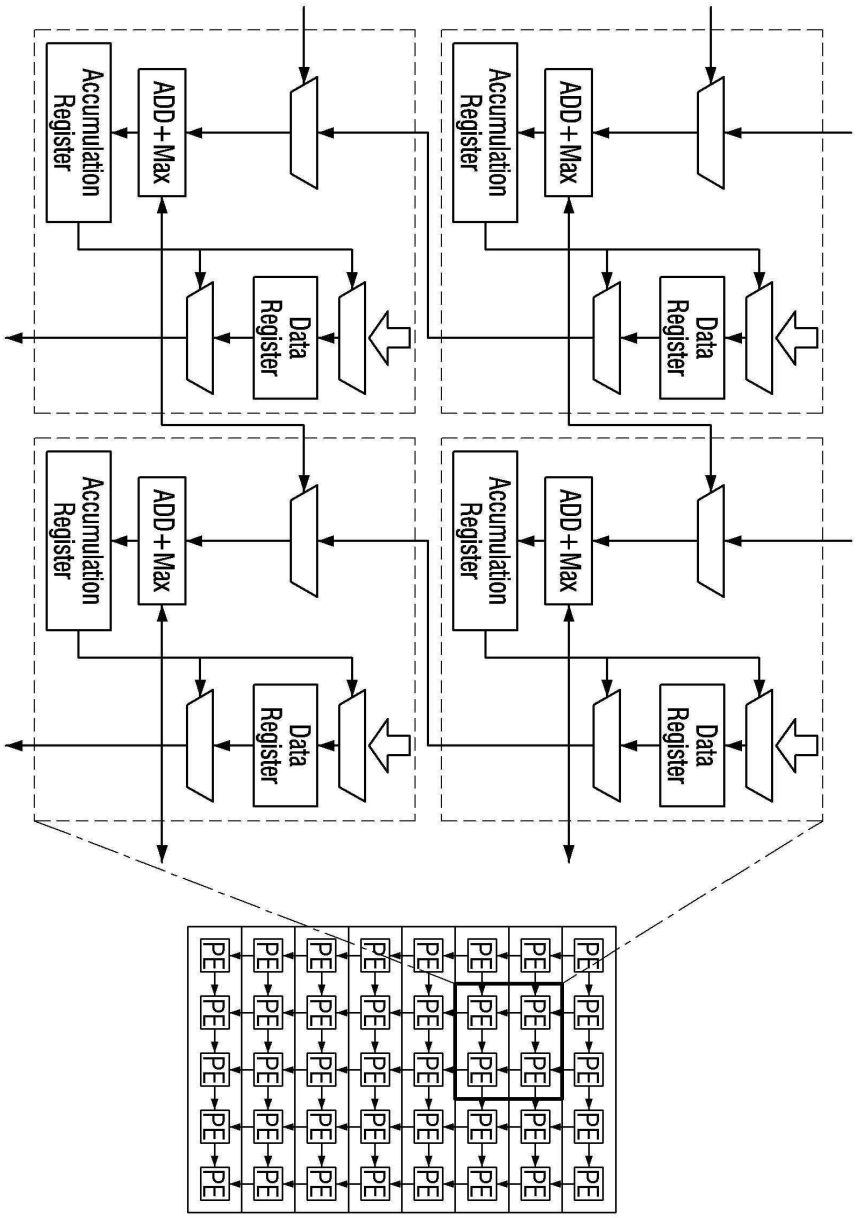
도면2b



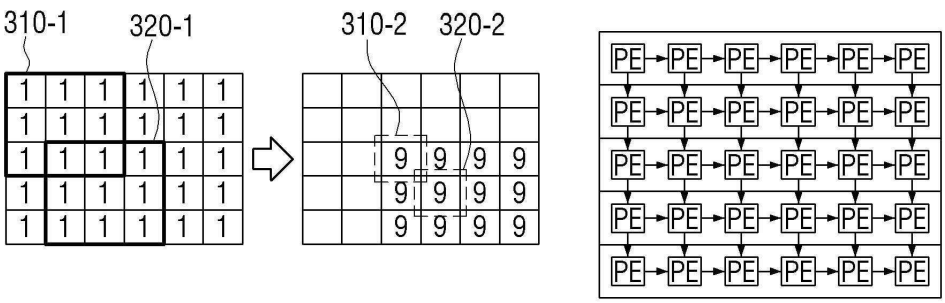
도면2c


243-1
240
241
ADD
1's comp.
242
243-2
Fetch from Mem.
230-1
230-3
Data Register
210
230-2
240
ADD+Max
220
Accumulation Register

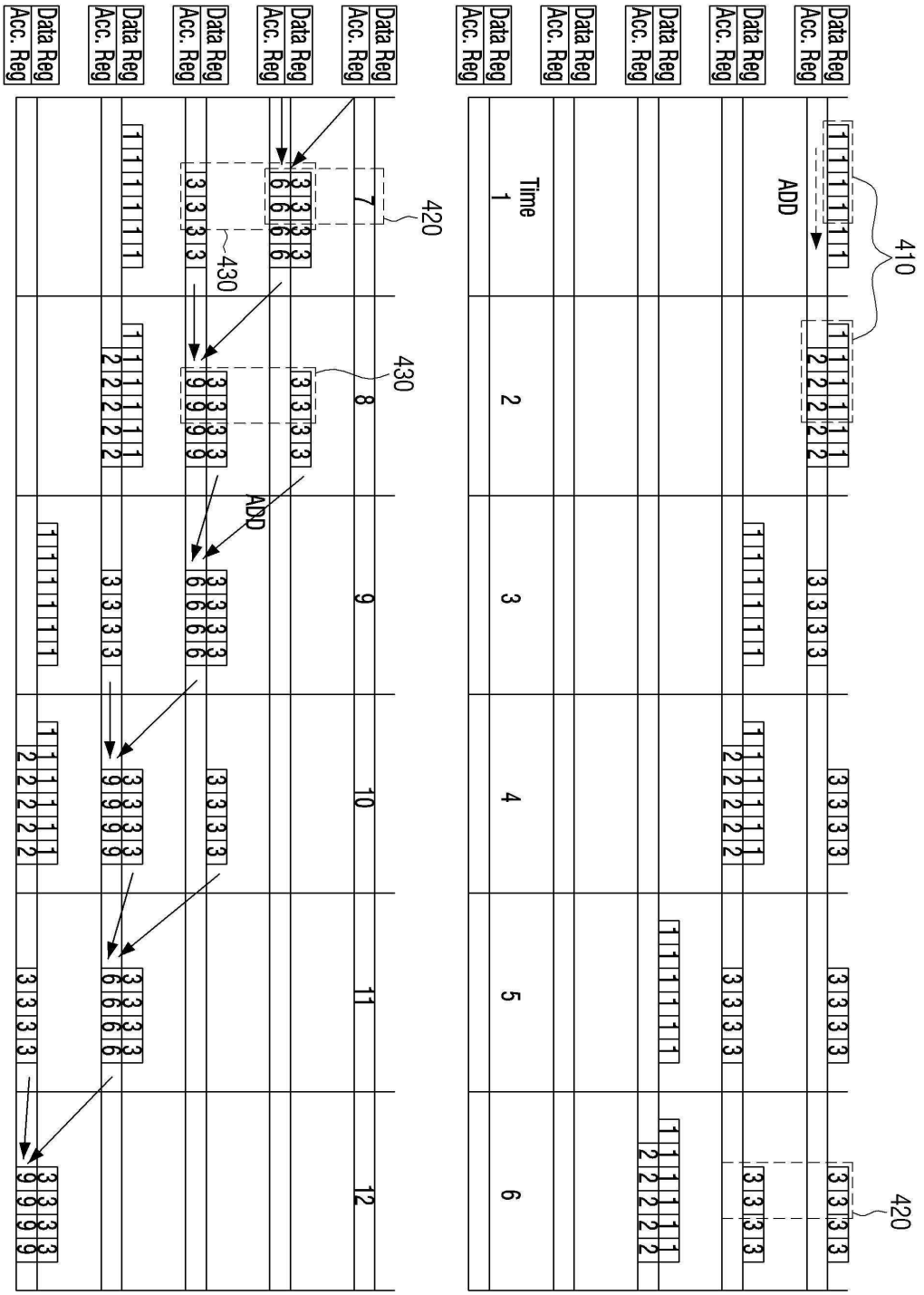
도면2d


Accumulation Register
ADD+Max
Data Register
PE

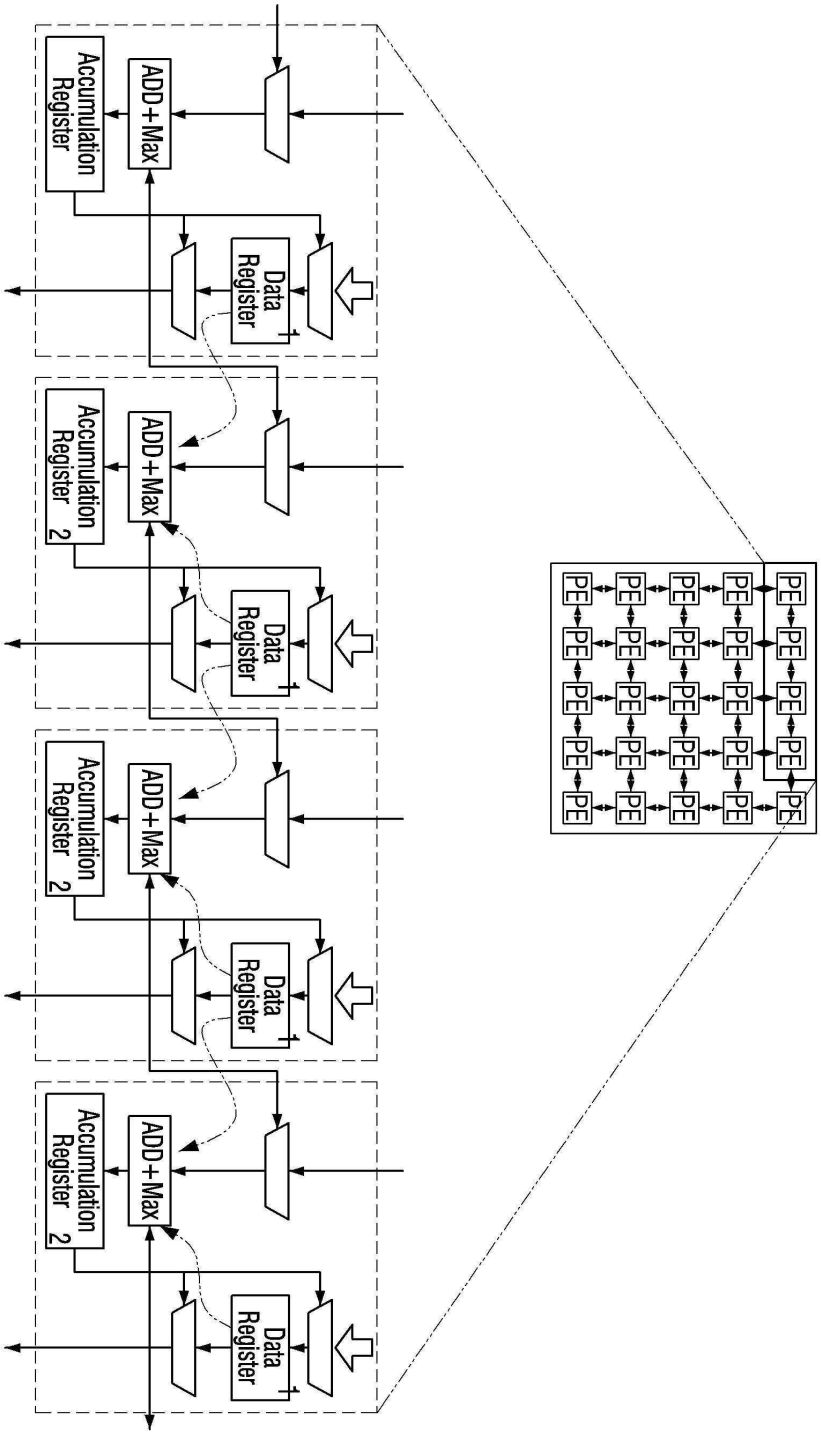
도면3


310-1
320-1
310-2
320-2
PE

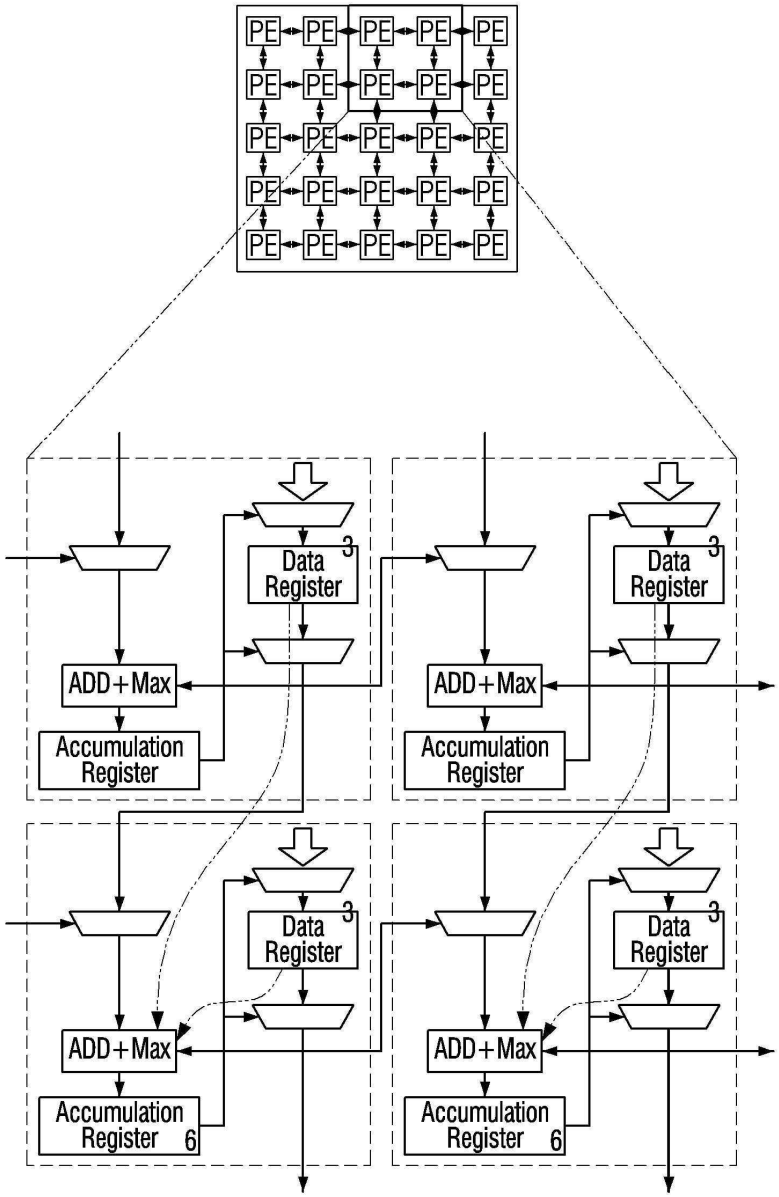
도면4



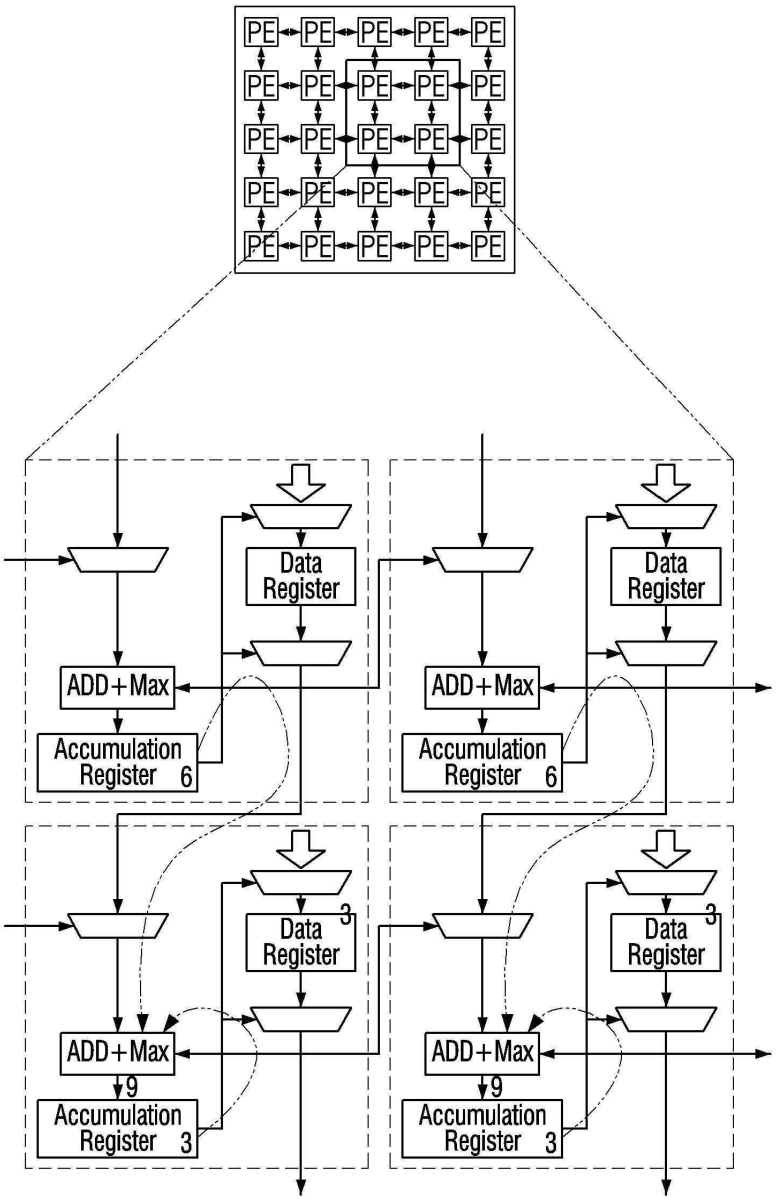
도면5


Accumulation Register
ADD+Max
Data Register
Accumulation Register 2
ADD+Max
Data Register
Accumulation Register 2
ADD+Max
Data Register
Accumulation Register 2
ADD+Max
Data Register
PE

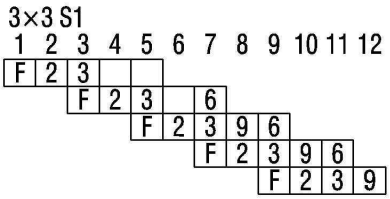
도면6


PE
Data Register
3
ADD+Max
Accumulation Register
Accumulation Register
6

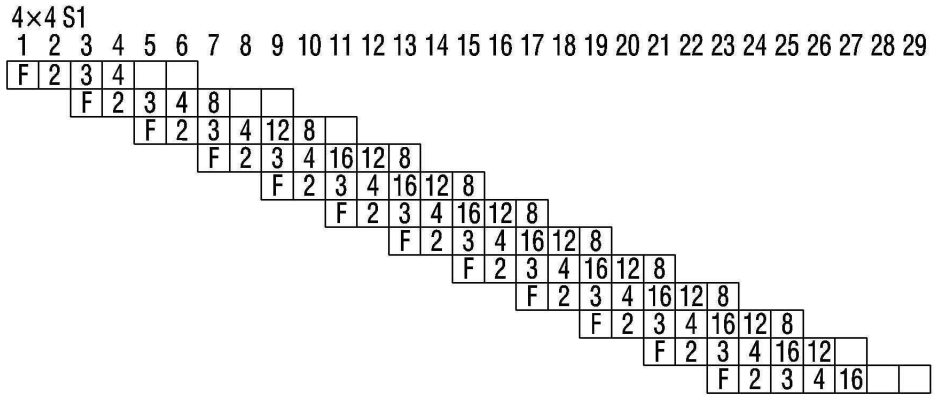
도면7


PE
Data Register
ADD+Max
Accumulation Register
6
3
9

도면8

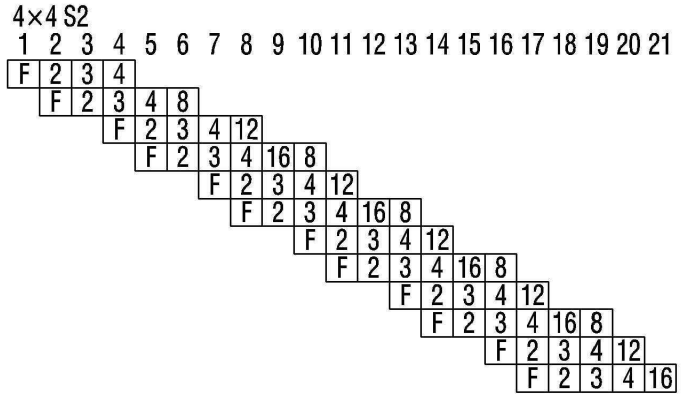

3×3 S1
1 2 3 4 5 6 7 8 9 10 11 12
F 2 3
F 2 3 6
F 2 3 9 6
F 2 3 9 6
F 2 3 9

도면9a



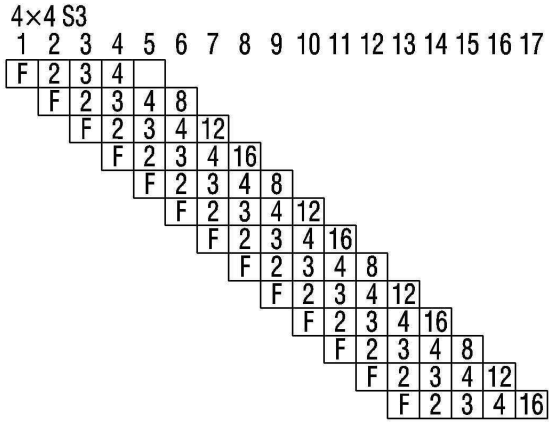
< 4×4 Pooling, Stride 1 >

도면9b



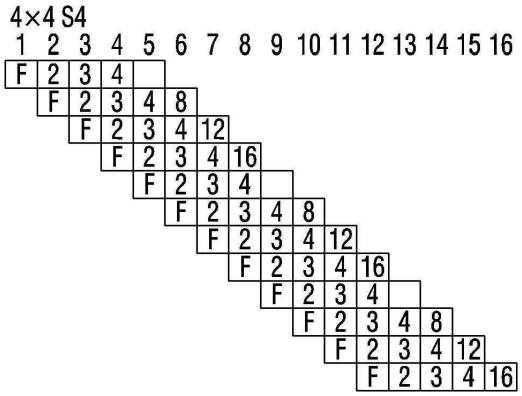
< 4×4 Pooling, Stride 2 >

도면9c



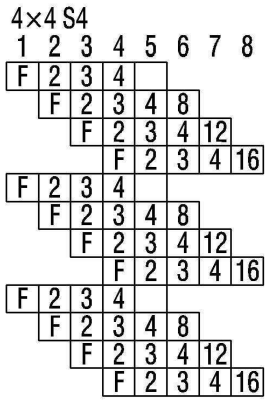
< 4×4 Pooling, Stride 3 >

도면9d



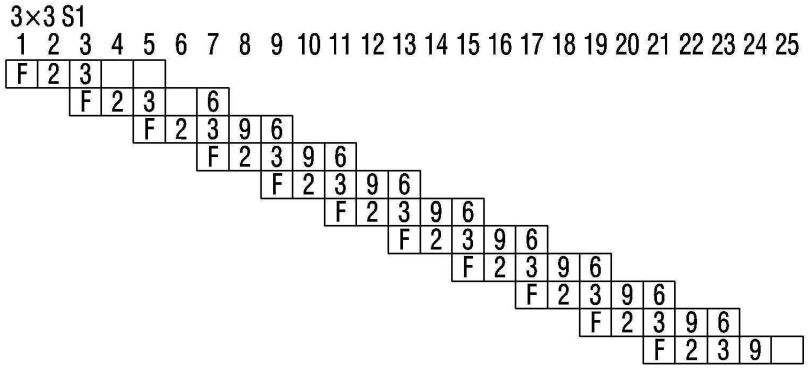
< 4×4 Pooling, Stride 4 >

도면9e



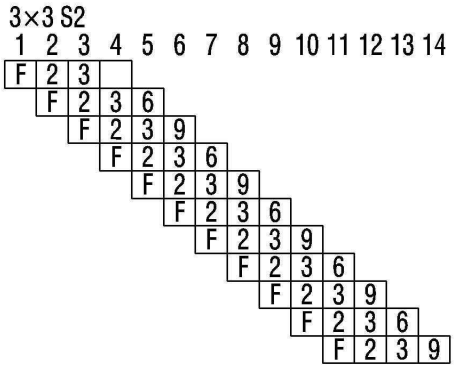
< 4×4 Pooling, Stride 4 >

도면10a



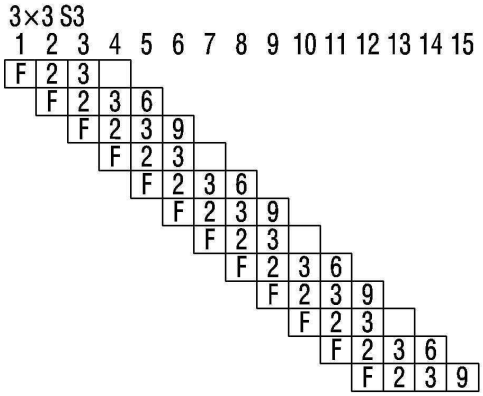
< 3×3 Pooling, Stride 1 >

도면10b



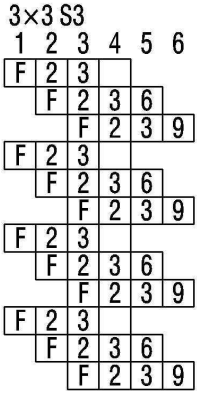
< 3×3 Pooling, Stride 2 >

도면10c



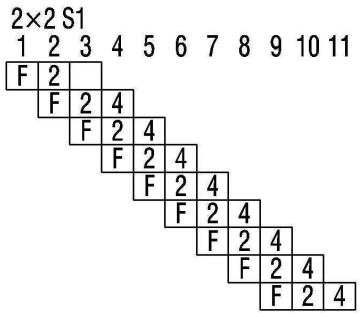
< 3×3 Pooling, Stride 3 >

도면10d



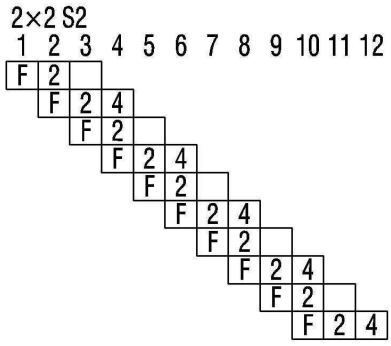
< 3×3 Pooling, Stride 3 >

도면11a



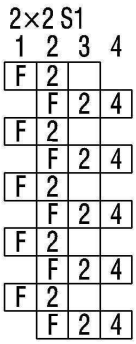
< 2×2 Pooling, Stride 1 >

도면11b



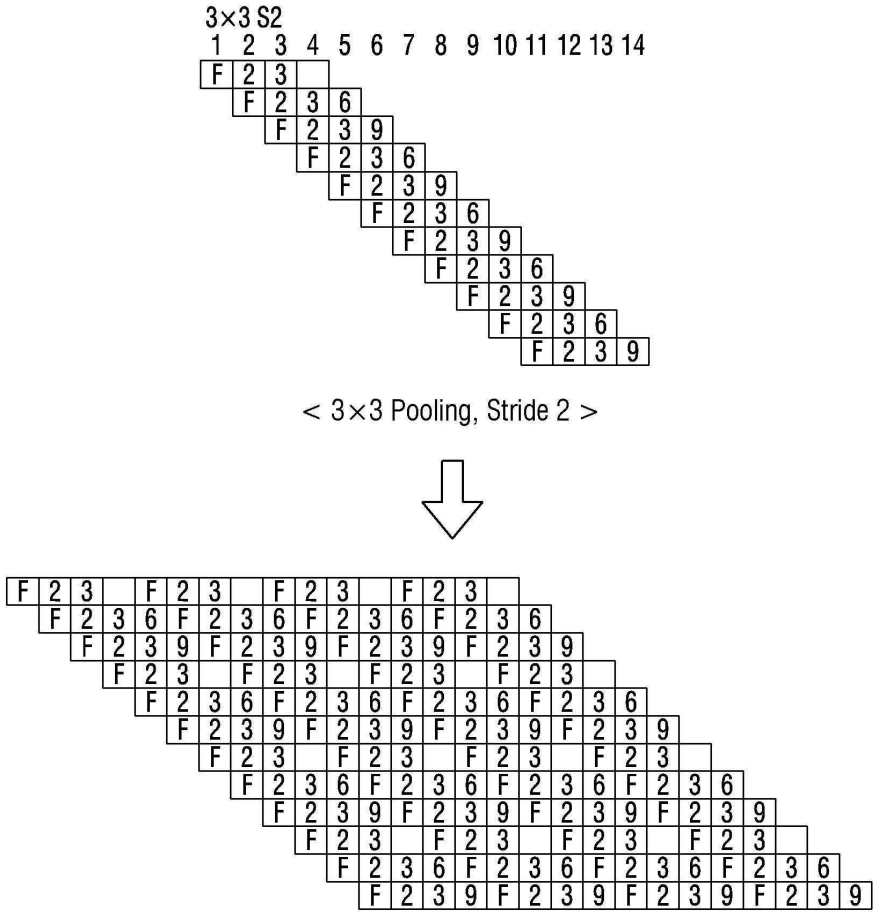
< 2×2 Pooling, Stride 2 >

도면11c

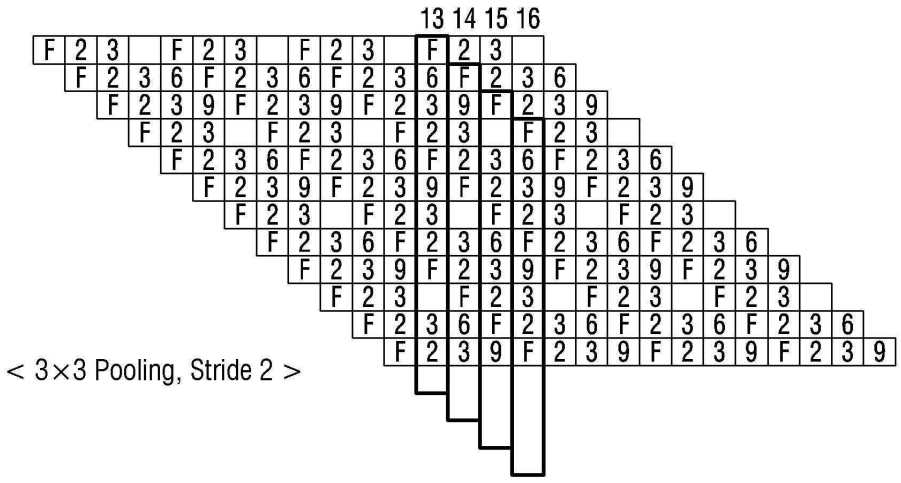


< 2×2 Pooling, Stride 2 >

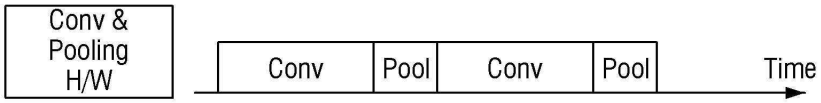
도면12



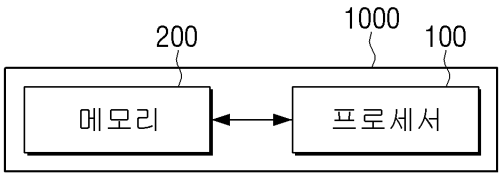
도면13



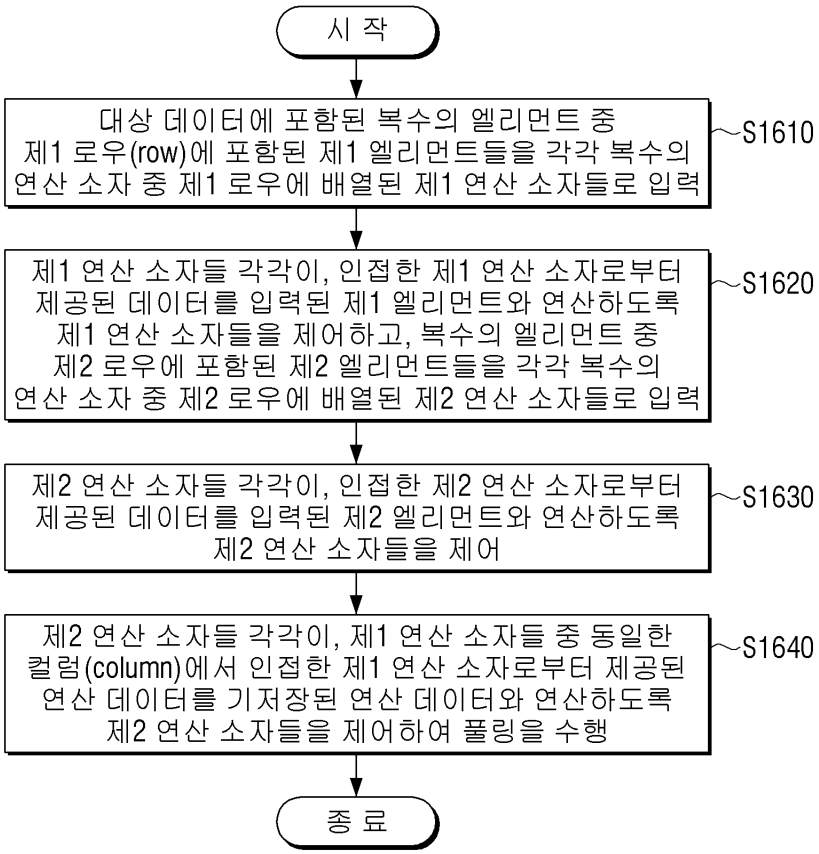
도면14



도면15


200
1000
100
메모리
프로세서

도면16


시 작
대상 데이터에 포함된 복수의 엘리먼트 중 제1 로우(row)에 포함된 제1 엘리먼트들을 각각 복수의 연산 소자 중 제1 로우에 배열된 제1 연산 소자들로 입력
S1610
제1 연산 소자들 각각이, 인접한 제1 연산 소자로부터 제공된 데이터를 입력된 제1 엘리먼트와 연산하도록 제1 연산 소자들을 제어하고, 복수의 엘리먼트 중 제2 로우에 포함된 제2 엘리먼트들을 각각 복수의 연산 소자 중 제2 로우에 배열된 제2 연산 소자들로 입력
S1620
제2 연산 소자들 각각이, 인접한 제2 연산 소자로부터 제공된 데이터를 입력된 제2 엘리먼트와 연산하도록 제2 연산 소자들을 제어
S1630
제2 연산 소자들 각각이, 제1 연산 소자들 중 동일한 컬럼(column)에서 인접한 제1 연산 소자로부터 제공된 연산 데이터를 기저장된 연산 데이터와 연산하도록 제2 연산 소자들을 제어하여 풀링을 수행
S1640
종 료