(12) INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
24 November 2016 (24.11.2016)

WIPO | PCT

(10) International Publication Number
WO 2016/186801 A1

(51) International Patent Classification:
G06N 3/02 (2006.01) *G06F 15/80* (2006.01)
G06N 3/063 (2006.01)

(21) International Application Number:
PCT/US2016/029294

(22) International Filing Date:
26 April 2016 (26.04.2016)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:
62/164,931 21 May 2015 (21.05.2015) US
14/844,524 3 September 2015 (03.09.2015) US

(71) Applicant: **GOOGLE INC.** [US/US]; 1600 Amphitheatre Parkway, Mountain View, California 94043 (US).

(72) Inventors: **ROSS, Jonathan**; 1600 Amphitheatre Parkway, Mountain View, California 94043 (US). **JOUPPI, Norman Paul**; 1600 Amphitheatre Parkway, Mountain View, California 94043 (US). **PHELPS, Andrew Everett**; 1600 Amphitheatre Parkway, Mountain View, California 94043 (US). **YOUNG, Reginald Clifford**; 1600 Amphitheatre Parkway, Mountain View, California 94043 (US). **NORRIE, Thomas**; 1600 Amphitheatre Parkway, Mountain View, California 94043 (US). **THORSON, Gregory Michael**; 1600 Amphitheatre Parkway, Mountain View, California 94043 (US). **LUU, Dan**; 1600 Amphitheatre Parkway, Mountain View, California 94043 (US).

(74) Agents: **HSIEH, I-Wei** et al.; Fish & Richardson P.C., P.O. Box 1022, Minneapolis, Minnesota 55440-1022 (US).

(81) Designated States *(unless otherwise indicated, for every kind of national protection available)*: AE, AG, AL, AM, AO, AT, AU, AZ, BA, BB, BG, BH, BN, BR, BW, BY, BZ, CA, CH, CL, CN, CO, CR, CU, CZ, DE, DK, DM, DO, DZ, EC, EE, EG, ES, FI, GB, GD, GE, GH, GM, GT, HN, HR, HU, ID, IL, IN, IR, IS, JP, KE, KG, KN, KP, KR, KZ, LA, LC, LK, LR, LS, LU, LY, MA, MD, ME, MG, MK, MN, MW, MX, MY, MZ, NA, NG, NI, NO, NZ, OM, PA, PE, PG, PH, PL, PT, QA, RO, RS, RU, RW, SA, SC, SD, SE, SG, SK, SL, SM, ST, SV, SY, TH, TJ, TM, TN, TR, TT, TZ, UA, UG, US, UZ, VC, VN, ZA, ZM, ZW.

(84) Designated States *(unless otherwise indicated, for every kind of regional protection available)*: ARIPO (BW, GH, GM, KE, LR, LS, MW, MZ, NA, RW, SD, SL, ST, SZ, TZ, UG, ZM, ZW), Eurasian (AM, AZ, BY, KG, KZ, RU,

[Continued on next page]

(54) Title: NEURAL NETWORK PROCESSOR

(57) Abstract: A circuit for performing neural network computations for a neural network comprising a plurality of neural network layers, the circuit comprising: a matrix computation unit configured to, for each of the plurality of neural network layers: receive a plurality of weight inputs and a plurality of activation inputs for the neural network layer, and generate a plurality of accumulated values based on the plurality of weight inputs and the plurality of activation inputs; and a vector computation unit communicatively coupled to the matrix computation unit and configured to, for each of the plurality of neural network layers: apply an activation function to each accumulated value generated by the matrix computation unit to generate a plurality of activated values for the neural network layer.

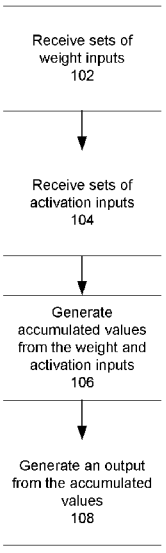

FIG. 1

WO 2016/186801 A1

TJ, TM), European (AL, AT, BE, BG, CH, CY, CZ, DE, DK, EE, ES, FI, FR, GB, GR, HR, HU, IE, IS, IT, LT, LU, LV, MC, MK, MT, NL, NO, PL, PT, RO, RS, SE, SI, SK, SM, TR), OAPI (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, KM, ML, MR, NE, SN, TD, TG).

Declarations under Rule 4.17:

— *as to applicant's entitlement to apply for and be granted a patent (Rule 4.17(ii))*

— *as to the applicant's entitlement to claim the priority of the earlier application (Rule 4.17(iii))*

Published:

— *with international search report (Art. 21(3))*

NEURAL NETWORK PROCESSOR

BACKGROUND

This specification relates to computing neural network inferences in hardware.

Neural networks are machine learning models that employ one or more layers of models to generate an output, e.g., a classification, for a received input. Some neural networks include one or more hidden layers in addition to an output layer. The output of each hidden layer is used as input to the next layer in the network, i.e., the next hidden layer or the output layer of the network. Each layer of the network generates an output from a received input in accordance with current values of a respective set of parameters.

SUMMARY

In general, this specification describes a special-purpose hardware circuit that computes neural network inferences.

In general, one innovative aspect of the subject matter described in this specification can be embodied in a circuit for performing neural network computations for a neural network comprising a plurality of neural network layers, the circuit comprising: a matrix computation unit configured to, for each of the plurality of neural network layers: receive a plurality of weight inputs and a plurality of activation inputs for the neural network layer, and generate a plurality of accumulated values based on the plurality of weight inputs and the plurality of activation inputs; and a vector computation unit communicatively coupled to the matrix computation unit and configured to, for each of the plurality of neural network layers: apply an activation function to each accumulated value generated by the matrix computation unit to generate a plurality of activated values for the neural network layer.

Implementations can include one or more of the following features. A unified buffer communicatively coupled to the matrix computation unit and the vector computation unit, where the unified buffer is configured to receive and store output from the vector computation unit, and the unified buffer is configured to send the received output as input to the matrix computation unit. A sequencer configured to receive instructions from a host device and generate a plurality of control signals from the instructions, where the plurality of control signals control dataflow through the circuit; and a direct memory access engine communicatively coupled to the unified buffer and the sequencer, where the direct memory access engine is configured to send the plurality of

activation inputs to the unified buffer, where the unified buffer is configured to send the plurality of activation inputs to the matrix computation unit, and where the direct memory access engine is configured to read result data from the unified buffer. A memory unit configured to send the plurality of weight inputs to the matrix computation unit, and where the direct memory access engine is configured to send the plurality of weight inputs to the memory unit. The matrix computation unit is configured as a two dimensional systolic array comprising a plurality of cells. The plurality of weight inputs is shifted through a first plurality of cells along a first dimension of the systolic array, and where the plurality of activation inputs is shifted through a second plurality of cells along a second dimension of the systolic array. For a given layer in the plurality of layers, a count of the plurality of activation inputs is greater than a size of the second dimension of the systolic array, and where the systolic array is configured to: divide the plurality of activation inputs into portions, where each portion has a size less than or equal to the size of the second dimension; generating, for each portion, a respective portion of accumulated values; and combining each portion of accumulated values to generate a vector of accumulated values for the given layer. For a given layer in the plurality of layers, a count of the plurality of weight inputs is greater than a size of the first dimension of the systolic array, and where the systolic array is configured to: divide the plurality of weight inputs into portions, where each portion has a size less than or equal to the size of the first dimension; generating, for each portion, a respective portion of accumulated values; and combining each portion of accumulated values to generate a vector of accumulated values for the given layer. Each cell in the plurality of cells comprises: a weight register configured to store a weight input; an activation register configured to store an activation input and configured to send the activation input to another activation register in a first adjacent cell along the second dimension; a sum-in register configured to store a previously summed value; multiplication circuitry communicatively coupled to the weight register and the activation register, where the multiplication circuitry is configured to output a product of the weight input and the activation input; and summation circuitry communicatively coupled to the multiplication circuitry and the sum-in register, where the summation circuitry is configured to output a sum of the product and the previously summed value, and where the summation circuitry is configured to send the sum to another sum-in register in a second adjacent cell along the first dimension. One or more cells in the plurality of cells are each configured to store the respective sum in a respective accumulator unit, where the respective sum is an accumulated value. The first

dimension of the systolic array corresponds to columns of the systolic array, and where the second dimension of the systolic array corresponds to rows of the systolic array. The vector computation unit normalizes each activated value to generate a plurality of normalized values. The vector computation unit pools one or more activated values to generate a plurality of pooled values.

Particular embodiments of the subject matter described in this specification can be implemented so as to realize one or more of the following advantages. Implementing a neural network processor in hardware improves efficiency, e.g., increase speed and throughput and reduce power and cost, over implementations in software. This can be useful for inference applications. Integrating components of the neural network processor into one circuit allows inferences to be computed without incurring penalties of off-chip communication. Additionally, the circuit can process neural network layers that have a number of inputs, e.g., a number of weight inputs or a number of activation inputs, larger than a size of a dimension of a matrix computation unit within the circuit. For example, the circuit can process a large number of weight inputs per neuron of the neural network.

The details of one or more embodiments of the subject matter of this specification are set forth in the accompanying drawings and the description below. Other features, aspects, and advantages of the subject matter will become apparent from the description, the drawings, and the claims.

BRIEF DESCRIPTION OF THE DRAWINGS

FIG. 1 is a flow diagram of an example method for performing a computation for a given layer of a neural network.

FIG. 2 shows an example neural network processing system.

FIG. 3 shows an example architecture including a matrix computation unit.

FIG. 4 shows an example architecture of a cell inside a systolic array.

FIG. 5 shows an example architecture of a vector computation unit.

FIG. 6 is a flow diagram of another example process for performing, using a systolic array, the computation for a given neural network layer having more activation inputs than rows in the systolic array.

Like reference numbers and designations in the various drawings indicate like elements.

DETAILED DESCRIPTION

A neural network having multiple layers can be used to compute inferences. For example, given an input, the neural network can compute an inference for the input. The neural network computes this inference by processing the input through each of the layers of the neural network. In particular, the layers of the neural network are arranged in a sequence, each with a respective set of weights. Each layer receives an input and processes the input in accordance with the set of weights for the layer to generate an output.

Therefore, in order to compute an inference from a received input, the neural network receives the input and processes it through each of the neural network layers in the sequence to generate the inference, with the output from one neural network layer being provided as input to the next neural network layer. Data inputs to a neural network layer, e.g., either the input to the neural network or the outputs of the layer below the layer in the sequence, to a neural network layer can be referred to as activation inputs to the layer.

In some implementations, the layers of the neural network are arranged in a directed graph. That is, any particular layer can receive multiple inputs, multiple outputs, or both. The layers of the neural network can also be arranged such that an output of a layer can be sent back as an input to a previous layer.

FIG. 1 is a flow diagram of an example process 100 for performing a computation for a given layer of a neural network using a special-purpose hardware circuit. For convenience, the method 100 will be described with respect to a system having one or more circuits that performs the method 100.The method 100 can be performed for each layer of the neural network in order to compute an inference from a received input.

The system receives sets of weight inputs (step 102) and sets of activation inputs (step 104) for the given layer. The sets of weight inputs and the sets of activation inputs can be received from dynamic memory and a unified buffer, respectively, of the special-purpose hardware circuit. In some implementations, both the sets of weight inputs and the sets of activation inputs can be received from the unified buffer.

The system generates accumulated values from the weight inputs and the activation inputs using a matrix multiplication unit of the special-purpose hardware circuit (step 106). In some implementations, the accumulated values are dot products of the sets of weight inputs and the sets of activation inputs. That is, for one set of weights, the system can multiply each weight input with each activation input and sum the

products together to form an accumulated value. The system can then compute dot products of other set of weights with other sets of activation inputs.

The system can generate a layer output from the accumulation values (step 108) using a vector computation unit of the special-purpose hardware circuit. In some implementations, the vector computation unit applies an activation function to the accumulated values, which will be described further below in reference to FIG. 5. The output of the layer can be stored in the unified buffer for use as an input to a subsequent layer in the neural network or can be used to determine the inference. The system finishes processing the neural network when a received input has been processed through each layer of the neural network to generate the inference for the received input.

FIG. 2 shows an example special-purpose integrated circuit 200 for performing neural network computations. The system 200 includes a host interface 202. The host interface 202 can receive instructions that include parameters for a neural network computation. The parameters can include at least one or more of the following: how many layers should be processed, corresponding sets of weight inputs for each layer of the layer, an initial set of activation inputs, i.e., the input to the neural network from which the inference is to be computed, corresponding input and output sizes of each layer, a stride value for the neural network computation, and a type of layer to be processed, e.g., a convolutional layer or a fully connected layer.

The host interface 202 can send the instructions to a sequencer 206, which converts the instructions into low level control signals that control the circuit to perform the neural network computations. In some implementations, the control signals regulate dataflow in the circuit, e.g., how the sets of weight inputs and the sets of activation inputs flow through the circuit. The sequencer 206 can send the control signals to a unified buffer 208, a matrix computation unit 212, and a vector computation unit 214. In some implementations, the sequencer 206 also sends control signals to a direct memory access engine 204 and dynamic memory 210. In some implementations, the sequencer 206 is a processor that generates clock signals. The sequencer 206 can use timing of the clock signals to, at appropriate times, send the control signals to each component of the circuit 200. In some other implementations, the host interface 202 passes in a clock signal from an external processor.

The host interface 202 can send the sets of weight inputs and the initial set of activation inputs to the direct memory access engine 204. The direct memory access engine 204 can store the sets of activation inputs at the unified buffer 208. In some

implementations, the direct memory access stores the sets of weights to dynamic memory 210, which can be a memory unit. In some implementations, the dynamic memory is located off of the circuit.

The unified buffer 208 is a memory buffer. It can be used to store the set of activation inputs from the direct memory access engine 204 and outputs of the vector computation unit 214. The vector computation unit will be described in more detail below with reference to FIG. 5. The direct memory access engine 204 can also read the outputs of the vector computation unit 214 from the unified buffer 208.

The dynamic memory 210 and the unified buffer 208 can send the sets of weight inputs and the sets of activation inputs, respectively, to the matrix computation unit 212. In some implementations, the matrix computation unit 212 is a two-dimensional systolic array. The matrix computation unit 212 can also be a one-dimensional systolic array or other circuitry that can perform mathematical operations, e.g., multiplication and addition. In some implementations, the matrix computation unit 212 is a general purpose matrix processor.

The matrix computation unit 212 can process the weight inputs and the activation inputs and provide a vector of outputs to the vector computation unit 214. In some implementations, the matrix computation unit sends the vector of outputs to the unified buffer 208, which sends the vector of outputs to the vector computation unit 214. The vector computation unit can process the vector of outputs and store a vector of processed outputs to the unified buffer 208. The vector of processed outputs can be used as activation inputs to the matrix computation unit 212, e.g., for use in a subsequent layer in the neural network. The matrix computation unit 212 and the vector computation unit 214 will be described in more detail below with reference to FIG. 3 and FIG. 5, respectively.

FIG. 3 shows an example architecture 300 including a matrix computation unit. The matrix computation unit is a two-dimensional systolic array 306. The two-dimensional systolic array 306 can be a square array. The array 306 includes multiple cells 304. In some implementations, a first dimension 320 of the systolic array 306 corresponds to columns of cells and a second dimension 322 of the systolic array 306 corresponds to rows of cells. The systolic array can have more rows than columns, more columns than rows, or an equal number of columns and rows.

In the illustrated example, value loaders 302 send activation inputs to rows of the array 306 and a weight fetcher interface 308 sends weight inputs to columns of the array

306. In some other implementations, however, activation inputs are transferred to the columns and weight inputs are transferred to the rows of the array 306.

The value loaders 302 can receive the activation inputs from a unified buffer, e.g., the unified buffer 208 of FIG. 2. Each value loader can send a corresponding activation input to a distinct left-most cell of the array 306. The left-most cell can be a cell along a left-most column of the array 306. For example, value loader 312 can send an activation input to cell 314. The value loader can also send the activation input to an adjacent value loader, and the activation input can be used at another left-most cell of the array 306. This allows activation inputs to be shifted for use in another particular cell of the array 306.

The weight fetcher interface 308 can receive the weight input from a memory unit, e.g., the dynamic memory 210 of FIG. 2. The weight fetcher interface 308 can send a corresponding weight input to a distinct top-most cell of the array 306. The top-most cell can be a cell along a top-most row of the array 306. For example, the weight fetcher interface 308 can send weight inputs to cells 314 and 316.

In some implementations, a host interface, e.g., the host interface 202 of FIG. 2, shifts activation inputs throughout the array 306 along one dimension, e.g., to the right, while shifting weight inputs throughout the array 306 along another dimension, e.g., to the bottom. For example, over one clock cycle, the activation input at cell 314 can shift to an activation register in cell 316, which is to the right of cell 314. Similarly, the weight input at cell 316 can shift to a weight register at cell 318, which is below cell 314.

On each clock cycle, each cell can process a given weight input and a given activation input to generate an accumulated output. The accumulated output can also be passed to an adjacent cell along the same dimension as the given weight input. An individual cell is described further below with reference FIG. 4.

The accumulated output can be passed along the same column as the weight input, e.g., towards the bottom of the column in the array 306. In some implementations, at the bottom of each column, the array 306 can include accumulator units 310 that store and accumulate each accumulated output from each column when performing calculations with layers having more weight inputs than columns or layers having more activation inputs than rows. In some implementations, each accumulator unit stores multiple parallel accumulations. This will be described further below with reference to FIG. 6. The accumulator units 310 can accumulate each accumulated output to generate a final accumulated value. The final accumulated value can be transferred to a vector

computation unit, e.g., the vector computation unit 502 of FIG. 5. In some other implementations, the accumulator units 310 passes the accumulated values to the vector computation unit without performing any accumulations when processing layers with fewer weight inputs than columns or layers having fewer activating inputs than rows.

FIG. 4 shows an example architecture 400 of a cell inside a systolic array, e.g., the systolic array 306 of FIG. 3.

The cell can include an activation register 406 that stores an activation input. The activation register can receive the activation input from a left adjacent cell, i.e., an adjacent cell located to the left of the given cell, or from a unified buffer, depending on the position of the cell within the systolic array. The cell can include a weight register 402 that stores a weight input. The weight input can be transferred from a top adjacent cell or from a weight fetcher interface, depending on the position of the cell within the systolic array. The cell can also include a sum in register 404. The sum in register 404 can store an accumulated value from the top adjacent cell. Multiplication circuitry 408 can be used to multiply the weight input from the weight register 402 with the activation input from the activation register 406. The multiplication circuitry 408 can output the product to summation circuitry 410.

The summation circuitry can sum the product and the accumulated value from the sum in register 404 to generate a new accumulated value. The summation circuitry 410 can then send the new accumulated value to another sum in register located in a bottom adjacent cell. The new accumulated value can be used as an operand for a summation in the bottom adjacent cell.

The cell can also shift the weight input and the activation input to adjacent cells for processing. For example, the weight register 402 can send the weight input to another weight register in the bottom adjacent cell. The activation register 406 can send the activation input to another activation register in the right adjacent cell. Both the weight input and the activation input can therefore be reused by other cells in the array at a subsequent clock cycle.

In some implementations, the cell also includes a control register. The control register can store a control signal that determines whether the cell should shift either the weight input or the activation input to adjacent cells. In some implementations, shifting the weight input or the activation input takes one or more clock cycles. The control signal can also determine whether the activation input or weight inputs are transferred to the multiplication circuitry 408, or can determine whether the multiplication circuitry 408

operates on the activation and weight inputs. The control signal can also be passed to one or more adjacent cells, e.g., using a wire.

In some implementations, weights are pre-shifted into a weight path register 412. The weight path register 412 can receive the weight input, e.g., from a top adjacent cell, and transfer the weight input to the weight register 402 based on the control signal. The weight register 402 can statically store the weight input such that as activation inputs are transferred to the cell, e.g., through the activation register 406, over multiple clock cycles, the weight input remains within the cell and is not transferred to an adjacent cell. Therefore, the weight input can be applied to multiple activation inputs, e.g., using the multiplication circuitry 408, and respective accumulated values can be transferred to an adjacent cell.

FIG. 5 shows an example architecture 500 of a vector computation unit 502. The vector computation unit 502 can receive a vector of accumulated values from a matrix computation unit, e.g., the matrix computation unit described in reference to FIG. 2.

The vector computation unit 502 can process the vector of accumulated values at the activation unit 404. In some implementations, the activation unit includes circuitry that applies a non-linear function to each accumulated value to generate activation values. For example, the non-linear function can be *tanh(x)*, where x is an accumulated value.

Optionally, the vector computation unit 502 can normalize the activation values in a normalization unit 506that generates normalized values from the activation values.

Also optionally, the vector computation unit 502 can pool values, either activation values or normalization values, using a pooling unit 508. The pooling unit 508 can apply an aggregation function to one or more of the normalized values to generate pooled values. In some implementations, the aggregation functions are functions that return a maximum, minimum, or average of the normalized values or of a subset of the normalized values.

Control signals 510 can be transferred, e.g., by the sequencer 206 of FIG. 2, and can regulate how the vector computation unit 502 processes the vector of accumulated values. That is, the control signals 510 can regulate whether the activation values are pooled, normalized, or both. The control signals 510 can also specify the activation, normalization, or pooling functions, as well as other parameters for normalization and pooling, e.g., a stride value.

The vector computation unit 502 can send values, e.g., activation values, normalized values, or pooled values, to a unified buffer, e.g., the unified buffer 208 of FIG. 2.

In some implementations, the pooling unit 508 receives the activation values instead of the normalization unit 506, and the pooling unit 508 sends the pooled values to the normalization unit 506, which generates normalized values to be stored in the unified buffer.

FIG. 6 is a flow diagram of example process for performing, using a systolic array, the computation for a given neural network layer having more activation inputs than rows in the systolic array. For convenience, the process 600 will be described with respect to a system that performs the process 600. In some implementations, a host interface or a sequencer performs the process 600, e.g., the host interface 202 or the sequencer 206, respectively, of FIG. 2. In some other implementations, the host interface receives instructions from an external processor that performs the process 600.

As described above, each layer can have multiple sets of activation inputs and each set of weight inputs can be transferred to cells at distinct rows of the array. In some implementations, some layers of the neural network have more sets of activation inputs than there are rows of the array.

The system can determine, e.g., using a comparator, whether there are more sets of activation inputs for the given neural network layer than there are rows in the systolic array. In some implementations, the system makes the determination at compile time. A set of activation inputs can correspond to the activation inputs provided to a single row of the array.

If there are more rows than sets of activation inputs (step 602), the system can generate accumulated values as described above in the systolic array 306 of FIG. 3 (step 604).

If there are more sets of activation inputs to be processed than there are rows in the array (step 602), the system can divide the sets of activation inputs into portions so that each portion has a size less than or equal to a number of rows in the array (step 606).

The system then can generate, for each portion of activation inputs, a portion of accumulated values (step 608). An accumulated value can be a sum of products of activation and weight inputs to cells along a given column, e.g., as described in systolic array 306 of FIG. 3. Each portion of accumulated values can be stored in a buffer until all portions of activation inputs have been processed. The buffer can be a buffer in

accumulator units 310 of FIG. 3, a buffer in the systolic array, or the unified buffer 208 of FIG. 2.

The system can then combine all portions of accumulated values into a vector of accumulated values (step 610). In particular, the system can access the buffer of previously stored portions of accumulated values and accumulate, e.g., using accumulator units 310 of FIG. 3, the accumulated values to generate a vector of the accumulated values. The system can send the vector of the accumulated values to a vector computation unit, e.g., the vector computation unit 214 of FIG. 2.

For example, if there are 256 rows in the array and there are 300 sets of activation inputs to process at a given layer, the system can generate 256 final accumulated values from 256 sets of activation inputs for complete utilization of the systolic array and store the 256 final accumulated values in a buffer. The system can then generate 44 final accumulated values from the 44 remainder sets of activation inputs. Finally, the system can combine all 300 final accumulated values to form a vector and send the vector to the vector computation unit.

If there are more sets of weight inputs than columns to the array, the system can perform similar operations. That is, the system can divide the sets of weight inputs into portions having fewer sets of weight inputs than a number of columns in the array, generate accumulated values for each portion, and combine the accumulated values into a vector for use in the vector computation unit. In some implementations, instead of comparing the number of sets of weight inputs with the number of columns in the array, the system can compare the number of accumulated values with the number of columns in the array.

Although the system has been described with weight inputs being transferred to columns of the array and activation inputs being transferred to rows of the array, in some implementations, the weight inputs are transferred to rows of the array and the activation inputs are transferred to columns of the array.

Although the hardware is described to be for computing inferences, the hardware can be used for one or more of the following: convolutional or fully-connected neural network training, linear or logistic regression, clustering, e.g., k-means clustering, video-encoding, and image processing.

Embodiments of the subject matter and the functional operations described in this specification can be implemented in digital electronic circuitry, in tangibly-embodied computer software or firmware, in computer hardware, including the structures disclosed

in this specification and their structural equivalents, or in combinations of one or more of them. Embodiments of the subject matter described in this specification can be implemented as one or more computer programs, i.e., one or more modules of computer program instructions encoded on a tangible non transitory program carrier for execution by, or to control the operation of, data processing apparatus. Alternatively or in addition, the program instructions can be encoded on an artificially generated propagated signal, e.g., a machine-generated electrical, optical, or electromagnetic signal, that is generated to encode information for transmission to suitable receiver apparatus for execution by a data processing apparatus. The computer storage medium can be a machine-readable storage device, a machine-readable storage substrate, a random or serial access memory device, or a combination of one or more of them.

The term "data processing apparatus" encompasses all kinds of apparatus, devices, and machines for processing data, including by way of example a programmable processor, a computer, or multiple processors or computers. The apparatus can include special purpose logic circuitry, e.g., an FPGA (field programmable gate array) or an ASIC (application specific integrated circuit). The apparatus can also include, in addition to hardware, code that creates an execution environment for the computer program in question, e.g., code that constitutes processor firmware, a protocol stack, a database management system, an operating system, or a combination of one or more of them.

A computer program (which may also be referred to or described as a program, software, a software application, a module, a software module, a script, or code) can be written in any form of programming language, including compiled or interpreted languages, or declarative or procedural languages, and it can be deployed in any form, including as a standalone program or as a module, component, subroutine, or other unit suitable for use in a computing environment. A computer program may, but need not, correspond to a file in a file system. A program can be stored in a portion of a file that holds other programs or data, e.g., one or more scripts stored in a markup language document, in a single file dedicated to the program in question, or in multiple coordinated files, e.g., files that store one or more modules, sub programs, or portions of code. A computer program can be deployed to be executed on one computer or on multiple computers that are located at one site or distributed across multiple sites and interconnected by a communication network.

The processes and logic flows described in this specification can be performed by one or more programmable computers executing one or more computer programs to

perform functions by operating on input data and generating output. The processes and logic flows can also be performed by, and apparatus can also be implemented as, special purpose logic circuitry, e.g., an FPGA (field programmable gate array) or an ASIC (application specific integrated circuit).

Computers suitable for the execution of a computer program include, by way of example, can be based on general or special purpose microprocessors or both, or any other kind of central processing unit. Generally, a central processing unit will receive instructions and data from a read only memory or a random access memory or both. The essential elements of a computer are a central processing unit for performing or executing instructions and one or more memory devices for storing instructions and data. Generally, a computer will also include, or be operatively coupled to receive data from or transfer data to, or both, one or more mass storage devices for storing data, e.g., magnetic, magneto optical disks, or optical disks. However, a computer need not have such devices. Moreover, a computer can be embedded in another device, e.g., a mobile telephone, a personal digital assistant (PDA), a mobile audio or video player, a game console, a Global Positioning System (GPS) receiver, or a portable storage device, e.g., a universal serial bus (USB) flash drive, to name just a few.

Computer readable media suitable for storing computer program instructions and data include all forms of nonvolatile memory, media and memory devices, including by way of example semiconductor memory devices, e.g., EPROM, EEPROM, and flash memory devices; magnetic disks, e.g., internal hard disks or removable disks; magneto optical disks; and CD ROM and DVD-ROM disks. The processor and the memory can be supplemented by, or incorporated in, special purpose logic circuitry.

To send for interaction with a user, embodiments of the subject matter described in this specification can be implemented on a computer having a display device, e.g., a CRT (cathode ray tube) or LCD (liquid crystal display) monitor, for displaying information to the user and a keyboard and a pointing device, e.g., a mouse or a trackball, by which the user can send input to the computer. Other kinds of devices can be used to send for interaction with a user as well; for example, feedback provided to the user can be any form of sensory feedback, e.g., visual feedback, auditory feedback, or tactile feedback; and input from the user can be received in any form, including acoustic, speech, or tactile input. In addition, a computer can interact with a user by sending documents to and receiving documents from a device that is used by the user; for

example, by sending web pages to a web browser on a user's client device in response to requests received from the web browser.

Embodiments of the subject matter described in this specification can be implemented in a computing system that includes a back end component, e.g., as a data server, or that includes a middleware component, e.g., an application server, or that includes a front end component, e.g., a client computer having a graphical user interface or a Web browser through which a user can interact with an implementation of the subject matter described in this specification, or any combination of one or more such back end, middleware, or front end components. The components of the system can be interconnected by any form or medium of digital data communication, e.g., a communication network. Examples of communication networks include a local area network ("LAN") and a wide area network ("WAN"), e.g., the Internet.

The computing system can include clients and servers. A client and server are generally remote from each other and typically interact through a communication network. The relationship of client and server arises by virtue of computer programs running on the respective computers and having a client-server relationship to each other.

While this specification contains many specific implementation details, these should not be construed as limitations on the scope of any invention or of what may be claimed, but rather as descriptions of features that may be specific to particular embodiments of particular inventions. Certain features that are described in this specification in the context of separate embodiments can also be implemented in combination in a single embodiment. Conversely, various features that are described in the context of a single embodiment can also be implemented in multiple embodiments separately or in any suitable subcombination. Moreover, although features may be described above as acting in certain combinations and even initially claimed as such, one or more features from a claimed combination can in some cases be excised from the combination, and the claimed combination may be directed to a subcombination or variation of a subcombination.

Similarly, while operations are depicted in the drawings in a particular order, this should not be understood as requiring that such operations be performed in the particular order shown or in sequential order, or that all illustrated operations be performed, to achieve desirable results. In certain circumstances, multitasking and parallel processing may be advantageous. Moreover, the separation of various system modules and components in the embodiments described above should not be understood as requiring

such separation in all embodiments, and it should be understood that the described program components and systems can generally be integrated together in a single software product or packaged into multiple software products.

Particular embodiments of the subject matter have been described. Other embodiments are within the scope of the following claims. For example, the actions recited in the claims can be performed in a different order and still achieve desirable results. As one example, the processes depicted in the accompanying figures do not necessarily require the particular order shown, or sequential order, to achieve desirable results. In certain implementations, multitasking and parallel processing may be advantageous.

CLAIMS

1. A circuit for performing neural network computations for a neural network comprising a plurality of neural network layers, the circuit comprising:

a matrix computation unit configured to, for each of the plurality of neural network layers:

receive a plurality of weight inputs and a plurality of activation inputs for the neural network layer,

generate a plurality of accumulated values based on the plurality of weight inputs and the plurality of activation inputs; and

a vector computation unit communicatively coupled to the matrix computation unit and configured to, for each of the plurality of neural network layers:

apply an activation function to each accumulated value generated by the matrix computation unit to generate a plurality of activated values for the neural network layer.

2. The circuit of claim 1, further comprising:

a unified buffer communicatively coupled to the matrix computation unit and the vector computation unit, where the unified buffer is configured to receive and store output from the vector computation unit, and the unified buffer is configured to send the received output as input to the matrix computation unit.

3. The circuit of claim 2, further comprising:

a sequencer configured to receive instructions from a host device and generate a plurality of control signals from the instructions, where the plurality of control signals control dataflow through the circuit; and

a direct memory access engine communicatively coupled to the unified buffer and the sequencer,

where the direct memory access engine is configured to send the plurality of activation inputs to the unified buffer,

where the unified buffer is configured to send the plurality of activation inputs to the matrix computation unit,

and where the direct memory access engine is configured to read result data from the unified buffer.

4. The circuit of claim 3, further comprising:

a memory unit configured to send the plurality of weight inputs to the matrix computation unit, and where the direct memory access engine is configured to send the plurality of weight inputs to the memory unit.

5. The circuit of any preceding claim, where the matrix computation unit is configured as a two dimensional systolic array comprising a plurality of cells.

6. The circuit of claim 5, where the two dimensional systolic array is a square array.

7. The circuit of claim 5 or 6, where the plurality of weight inputs is shifted through a first plurality of cells along a first dimension of the systolic array, and where the plurality of activation inputs is shifted through a second plurality of cells along a second dimension of the systolic array.

8. The circuit of claim 7, where, for a given layer in the plurality of layers, a count of the plurality of activation inputs is greater than a size of the second dimension of the systolic array, and where the systolic array is configured to:

divide the plurality of activation inputs into portions, where each portion has a size less than or equal to the size of the second dimension;

generating, for each portion, a respective portion of accumulated values; and

combining each portion of accumulated values to generate a vector of accumulated values for the given layer.

9. The circuit of claim 7 or 8, where, for a given layer in the plurality of layers, a count of the plurality of weight inputs is greater than a size of the first dimension of the systolic array, and where the systolic array is configured to:

divide the plurality of weight inputs into portions, where each portion has a size less than or equal to the size of the first dimension;

generating, for each portion, a respective portion of accumulated values; and

combining each portion of accumulated values to generate a vector of accumulated values for the given layer.

10. The circuit of any of claims 7 to 9, where each cell in the plurality of cells comprises:

a weight register configured to store a weight input;

an activation register configured to store an activation input and configured to send the activation input to another activation register in a first adjacent cell along the second dimension;

a sum-in register configured to store a previously summed value;

multiplication circuitry communicatively coupled to the weight register and the activation register, where the multiplication circuitry is configured to output a product of the weight input and the activation input; and

summation circuitry communicatively coupled to the multiplication circuitry and the sum-in register, where the summation circuitry is configured to output a sum of the product and the previously summed value, and where the summation circuitry is configured to send the sum to another sum-in register in a second adjacent cell along the first dimension.

11. The circuit of claim 10, where one or more cells in the plurality of cells are each configured to store the respective sum in a respective accumulator unit, where the respective sum is an accumulated value.

12. The circuit of any of claims 7 to 11, where the first dimension of the systolic array corresponds to columns of the systolic array, and where the second dimension of the systolic array corresponds to rows of the systolic array.

13. The circuit of any preceding claim, where the vector computation unit normalizes each activated value to generate a plurality of normalized values.

14. The circuit of any preceding claim, where the vector computation unit pools one or more activated values to generate a plurality of pooled values.

15. A method for performing neural network computations for a neural network comprising a plurality of neural network layers using a circuit comprising a matrix computation unit and a vector computation unit coupled to the matrix computation unit, the method comprising, for each of the plurality of neural network layers:

providing a plurality of weight inputs and a plurality of activation inputs for the neural network layer to the matrix computation unit;

generating, using the matrix computation unit, a plurality of accumulated values, wherein the matrix computation unit is configured to receive the plurality of weight inputs and the plurality of activation inputs for the neural network layer and generate the plurality of accumulated values based on the plurality of weight inputs and the plurality of activation inputs; and

generating, using the vector computation unit, a plurality of activated values for the neural network layer, wherein the matrix computation unit is configured to apply an activation function to each accumulated value generated by the matrix computation unit to generate a plurality of activated values for the neural network layer.

16. The method of claim 15, further comprising:

receiving, by a unified buffer communicatively coupled to the matrix computation unit and the vector computation unit;

storing output from the vector computation unit at the unified buffer;

sending, from the unified buffer, the received output as input to the matrix computation unit.

17. The method of claim 16, further comprising:

receiving, at a sequencer, instructions from a host device and generating a plurality of control signals from the instructions, where the plurality of control signals control dataflow through the circuit;

sending, from a direct memory access engine communicatively coupled to the unified buffer and the sequencer, the plurality of activation inputs to the unified buffer;

sending, from the unified buffer, the plurality of activation inputs to the matrix computation unit; and

reading, at the direct memory access engine, result data from the unified buffer.

18. The method of claim 17, further comprising:

sending, at a memory unit, the plurality of weight inputs to the matrix computation unit;

sending, from the direct memory access engine, the plurality of weight inputs to the memory unit.

19. The method of any of claims 15 to 18, where the matrix computation unit is configured as a two dimensional systolic array comprising a plurality of cells.

20. The method of claim 19, where the two dimensional systolic array is a square array.

21. The method of claim 19 or 20, where the plurality of weight inputs is shifted through a first plurality of cells along a first dimension of the systolic array, and where the plurality of activation inputs is shifted through a second plurality of cells along a second dimension of the systolic array.

22. The method of claim 21, where, for a given layer in the plurality of layers, a count of the plurality of activation inputs is greater than a size of the second dimension of the systolic array, the method further comprising:

dividing, at the systolic array, the plurality of activation inputs into portions, where each portion has a size less than or equal to the size of the second dimension;

generating, for each portion at the systolic array, a respective portion of accumulated values; and

combining, at the systolic array, each portion of accumulated values to generate a vector of accumulated values for the given layer.

23. The method of claim 21 or 22, where, for a given layer in the plurality of layers, a count of the plurality of weight inputs is greater than a size of the first dimension of the systolic array, the method further comprising:

dividing, at the systolic array, the plurality of weight inputs into portions, where each portion has a size less than or equal to the size of the first dimension;

generating, for each portion at the systolic array, a respective portion of accumulated values; and

combining, at the systolic array, each portion of accumulated values to generate a vector of accumulated values for the given layer.

24. The method of any of claims 21 to 23, where each cell in the plurality of cells comprises:

a weight register configured to store a weight input;

an activation register configured to store an activation input and configured to send the activation input to another activation register in a first adjacent cell along the second dimension;

a sum-in register configured to store a previously summed value;

multiplication circuitry communicatively coupled to the weight register and the activation register, where the multiplication circuitry is configured to output a product of the weight input and the activation input; and

summation circuitry communicatively coupled to the multiplication circuitry and the sum-in register, where the summation circuitry is configured to output a sum of the product and the previously summed value, and where the summation circuitry is configured to send the sum to another sum-in register in a second adjacent cell along the first dimension.

25. The method of claim 24, further comprising storing, at one or more cells in the plurality of cells, the respective sum in a respective accumulator unit, where the respective sum is an accumulated value.

26. The method of any of claims 21 to 25, where the first dimension of the systolic array corresponds to columns of the systolic array, and where the second dimension of the systolic array corresponds to rows of the systolic array.

27. The method of any of claims 15 to 26, further comprising normalizing, at the vector computation unit, each activated value to generate a plurality of normalized values.

28. The method of any of claims 15 to 26, further comprising pooling, at the vector computation unit, one or more activated values to generate a plurality of pooled values.

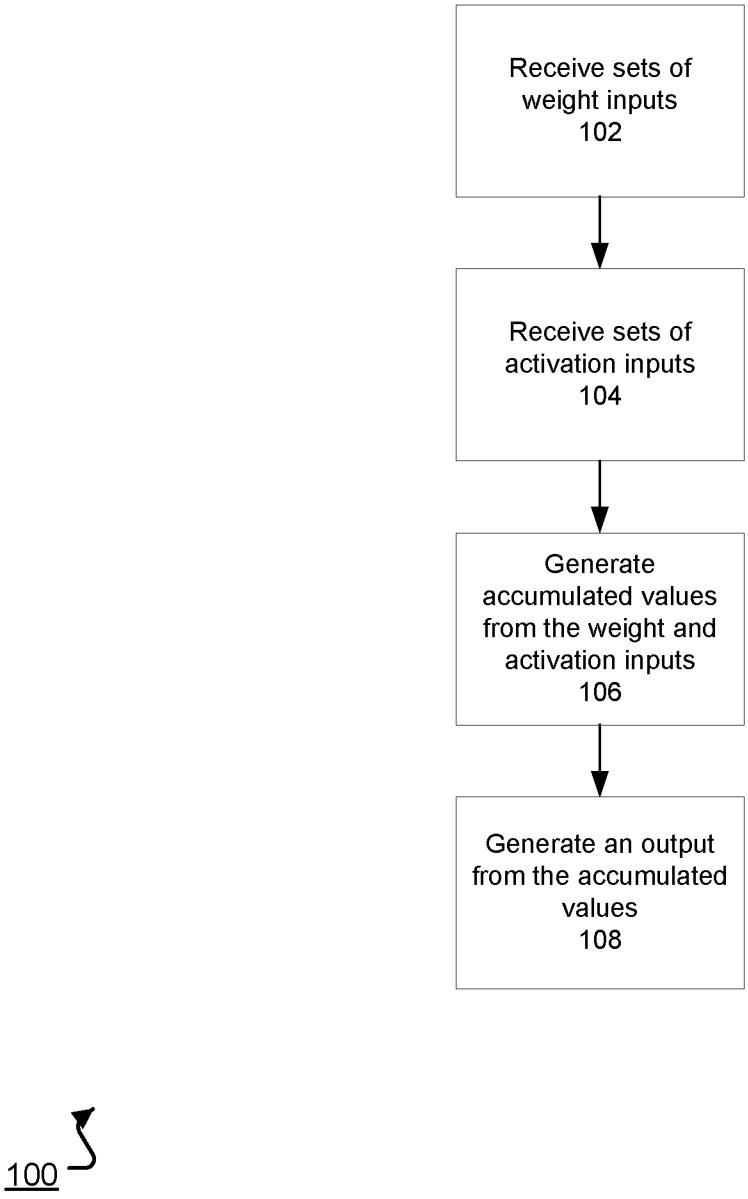
Receive sets of weight inputs 102
Receive sets of activation inputs 104
Generate accumulated values from the weight and activation inputs 106
Generate an output from the accumulated values 108
100

FIG. 1

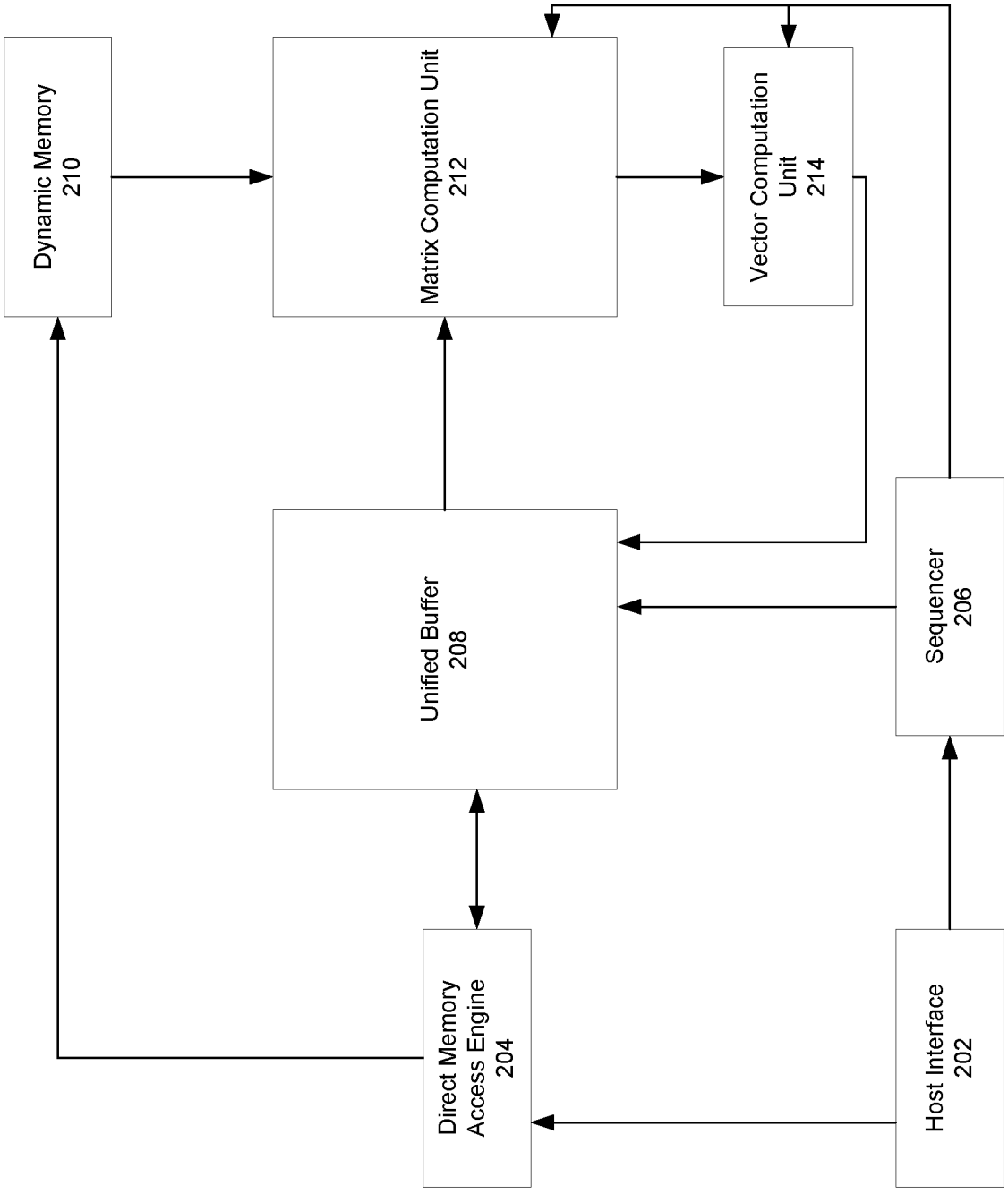
Dynamic Memory
210
Matrix Computation Unit
212
Vector Computation
Unit
214
Unified Buffer
208
Sequencer
206
Direct Memory
Access Engine
204
Host Interface
202
200

FIG. 2

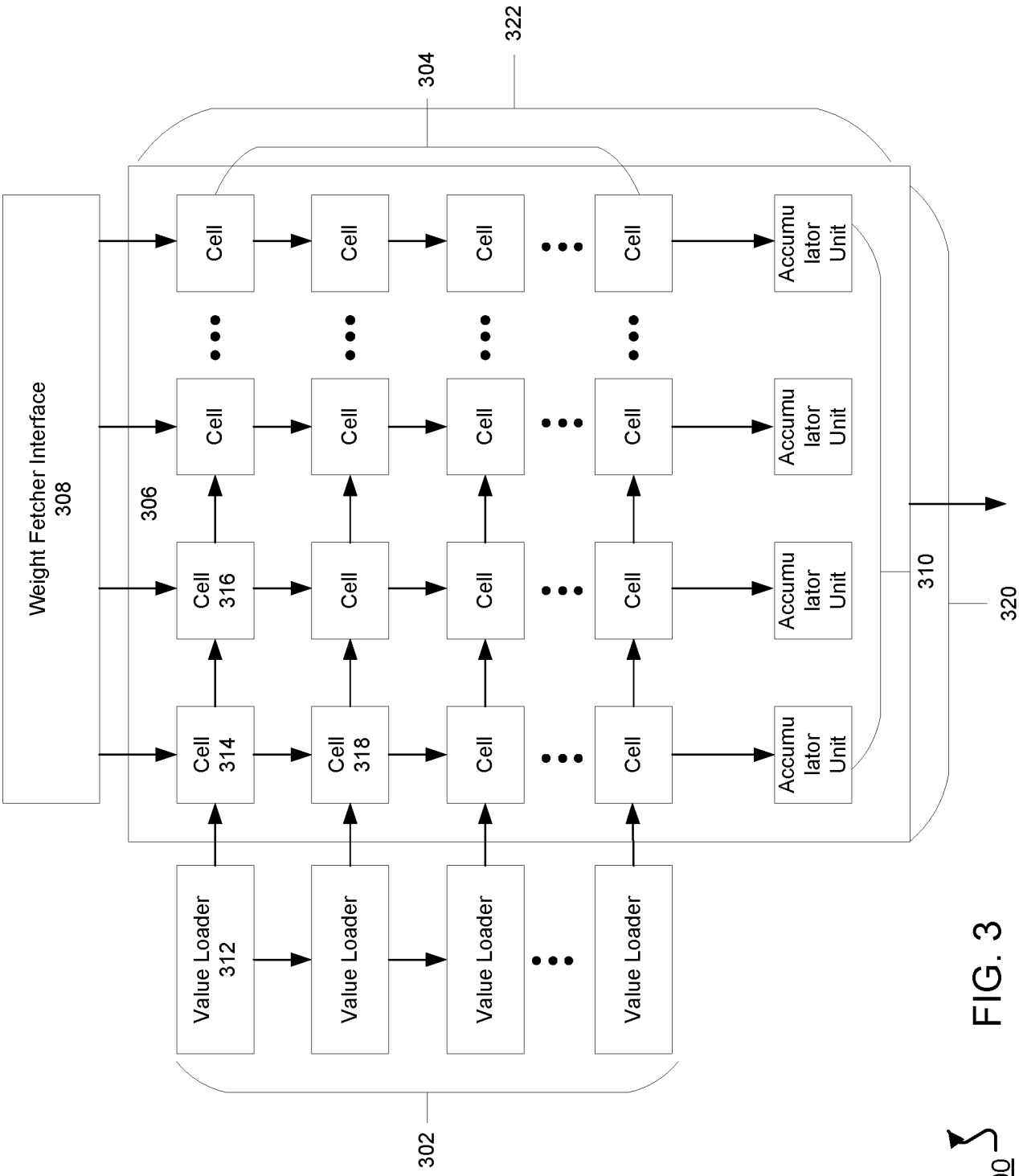
Weight Fetcher Interface
308
306
Value Loader
312
Cell
314
Cell
316
Cell
318
Cell
Accumu
lator
Unit
304
322
302
310
320
300

FIG. 3

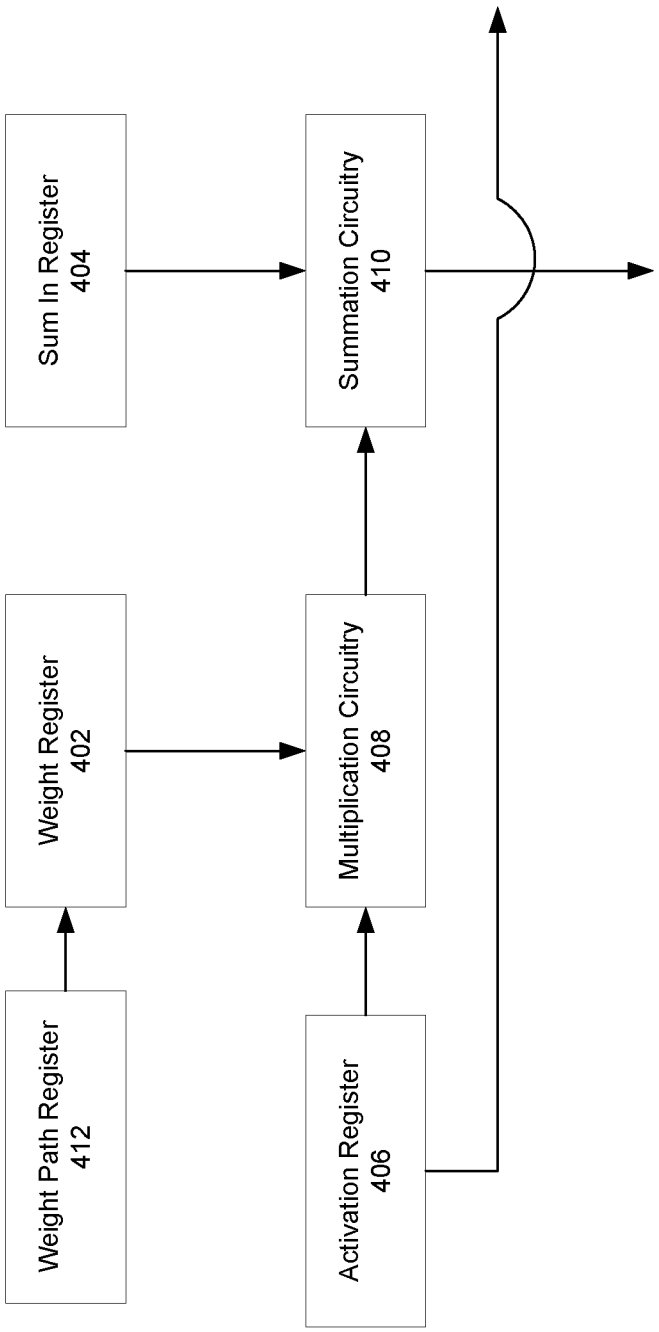
Sum In Register
404
Summation Circuitry
410
Weight Register
402
Multiplication Circuitry
408
Weight Path Register
412
Activation Register
406
400

FIG. 4

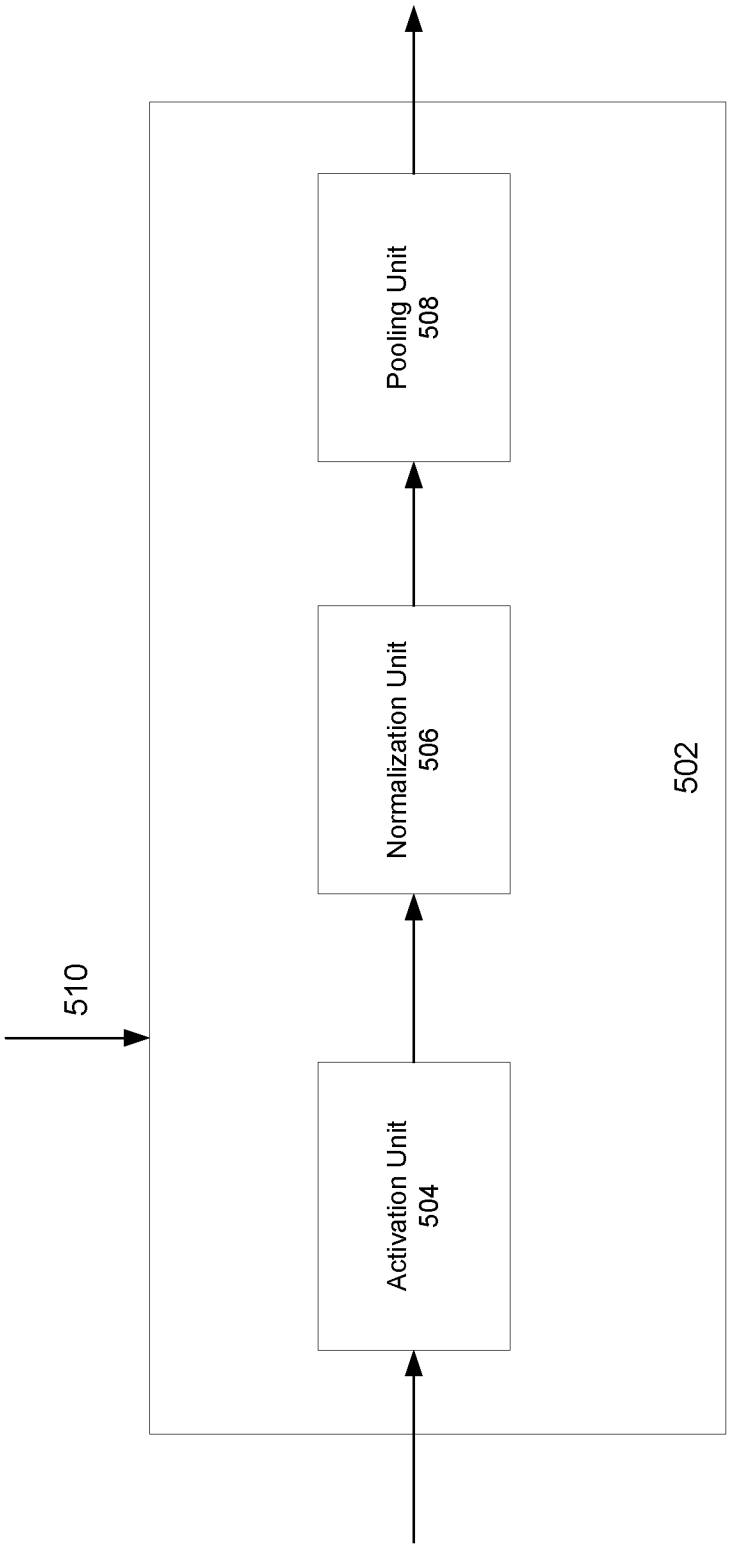
510
Activation Unit
504
Normalization Unit
506
Pooling Unit
508
502
500

FIG. 5

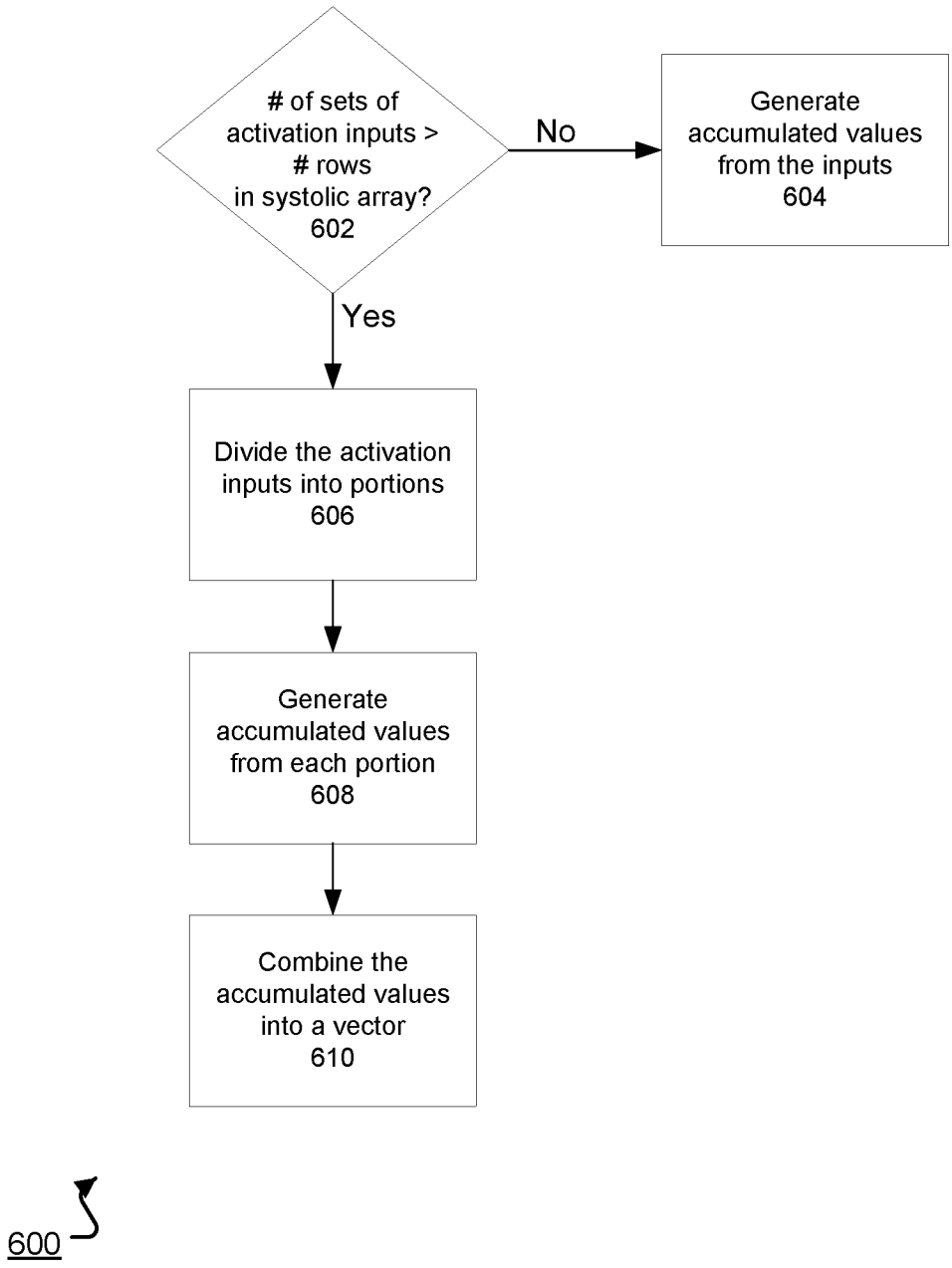
# of sets of activation inputs > # rows in systolic array? 602
No
Generate accumulated values from the inputs 604
Yes
Divide the activation inputs into portions 606
Generate accumulated values from each portion 608
Combine the accumulated values into a vector 610
600

FIG. 6

INTERNATIONAL SEARCH REPORT

International application No
PCT/US2016/029294

A. CLASSIFICATION OF SUBJECT MATTER
INV. G06N3/02 G06N3/063 G06F15/80
ADD.

According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)
G06N G06F

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched

Electronic data base consulted during the international search (name of data base and, where practicable, search terms used)
EPO-Internal, WPI Data

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
X	EP 0 422 348 A2 (HNC INC [US]) 17 April 1991 (1991-04-17) abstract page 2, line 4 - line 31 page 3, line 21 - page 4, line 33 page 4, line 55 - page 5, line 31 page 6, line 54 - line 56 page 7, line 18 - line 24 page 7, line 36 - line 42 page 8, line 32 - page 9, line 3 figures 1,3,4A,10 ----- -/--	1-28

[X] Further documents are listed in the continuation of Box C.

[X] See patent family annex.

* Special categories of cited documents :

"A" document defining the general state of the art which is not considered to be of particular relevance

"E" earlier application or patent but published on or after the international filing date

"L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)

"O" document referring to an oral disclosure, use, exhibition or other means

"P" document published prior to the international filing date but later than the priority date claimed

"T" later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention

"X" document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone

"Y" document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art

"&" document member of the same patent family

Date of the actual completion of the international search	Date of mailing of the international search report
25 August 2016	01/09/2016

Name and mailing address of the ISA/	Authorized officer
European Patent Office, P.B. 5818 Patentlaan 2 NL - 2280 HV Rijswijk Tel. (+31-70) 340-2040, Fax: (+31-70) 340-3016	Fantini, Federico

INTERNATIONAL SEARCH REPORT

International application No
PCT/US2016/029294

C(Continuation). DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
X	US 2014/180989 A1 (KRIZHEVSKY ALEXANDER [CA] ET AL) 26 June 2014 (2014-06-26) abstract paragraph [0002] - paragraph [0003] paragraph [0009] - paragraph [0010] paragraph [0022] - paragraph [0032] paragraph [0039] figures 1,2,4	1-28

X	KUNG S: "VLSI Array processors", IEEE ASSP MAGAZINE, IEEE, US, vol. 2, no. 3, 1 July 1985 (1985-07-01), pages 4-22, XP011370547, ISSN: 0740-7467, DOI: 10.1109/MASSP.1985.1163741 paragraph [02.2] paragraph [03.1] paragraph [03.4] paragraph [05.2] - paragraph [05.3] figure 2.4 figure 3.1 figure 5.1	1-28

X	US 2011/029471 A1 (CHAKRADHAR SRIMAT [US] ET AL) 3 February 2011 (2011-02-03) abstract paragraph [0005] - paragraph [0010] paragraph [0012] paragraph [0025] paragraph [0030] - paragraph [0036] paragraph [0038] - paragraph [0039] paragraph [0043] - paragraph [0044] paragraph [0053] - paragraph [0054] paragraph [0056] paragraph [0059] paragraph [0064] - paragraph [0066] paragraph [0071] figures 1,4	1-28

A	US 8 924 455 B1 (BARMAN KAUSHIK [IN] ET AL) 30 December 2014 (2014-12-30) abstract column 1, line 12 - line 24 column 2, line 65 - column 5, line 15 column 6, line 46 - line 59 column 7, line 11 - line 25 column 8, line 42 - line 61 figures 13,15	1-28

	-/--	

INTERNATIONAL SEARCH REPORT

International application No
PCT/US2016/029294

C(Continuation). DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
A	Alex Krizhevsky ET AL: "ImageNet classification with deep convolutional neural networks", The 26th annual conference on Neural Information Processing Systems (NIPS'25): 3-8 December 2012, 6 December 2012 (2012-12-06), XP055113686, Retrieved from the Internet: URL:http://books.nips.cc/papers/files/nips25/NIPS2012_0534.pdf [retrieved on 2014-04-11] abstract page 1 - page 2, paragraph 1 page 2, paragraph 3 - page 5, paragraph 4 figure 2	1-28
A	US 2014/288928 A1 (PENN GERALD BRADLEY [CA] ET AL) 25 September 2014 (2014-09-25) abstract paragraph [0015] - paragraph [0017] paragraph [0019] paragraph [0023] - paragraph [0027] paragraph [0029] - paragraph [0031] figures 1,2	1-28
A	US 2014/337262 A1 (KATO MASAMI [JP] ET AL) 13 November 2014 (2014-11-13) paragraph [0004] - paragraph [0006] paragraph [0009] paragraph [0014] paragraph [0042] - paragraph [0056] paragraph [0066] paragraph [0084] - paragraph [0085] paragraph [0096] figures 1,2,13	1-28
A	US 2007/086655 A1 (SIMARD PATRICE Y [US] ET AL) 19 April 2007 (2007-04-19) abstract paragraph [0026] - paragraph [0028] paragraph [0030] - paragraph [0033] paragraph [0035] - paragraph [0039] paragraph [0046] paragraph [0051] paragraph [0053] - paragraph [0054]	1-28

INTERNATIONAL SEARCH REPORT

Information on patent family members

International application No
PCT/US2016/029294

Patent document cited in search report	Publication date	Patent family member(s)	Publication date
EP 0422348 A2	17-04-1991	EP 0422348 A2	17-04-1991
		JP H03131965 A	05-06-1991
		US 5138695 A	11-08-1992
		US 5471627 A	28-11-1995
US 2014180989 A1	26-06-2014	AU 2013370514 A1	16-07-2015
		TW 201439926 A	16-10-2014
		US 2014180989 A1	26-06-2014
		US 2015339571 A1	26-11-2015
		WO 2014105865 A1	03-07-2014
US 2011029471 A1	03-02-2011	NONE	
US 8924455 B1	30-12-2014	NONE	
US 2014288928 A1	25-09-2014	US 2014288928 A1	25-09-2014
		US 2015255062 A1	10-09-2015
US 2014337262 A1	13-11-2014	JP 5376920 B2	25-12-2013
		JP 2010134697 A	17-06-2010
		US 2011239032 A1	29-09-2011
		US 2014337262 A1	13-11-2014
		WO 2010064728 A1	10-06-2010
US 2007086655 A1	19-04-2007	NONE	