US Patent & Trademark Office
Patent Public Search | Text View

United States Patent Application Publication	20250258889
Kind Code	A1
Publication Date	August 14, 2025
Inventor(s)	KIM; Ho Young

CONVOLUTION PROCESSING METHOD AND ELECTRONIC APPARATUS PERFORMING THE SAME

Abstract

A convolution processing method performed by at least one processor included in an electronic apparatus, including: obtaining a plurality of partial sums by multiplying a plurality of input feature values in a binary form by a plurality of weight values in the binary form; inverting a plurality of first partial sums from among the plurality of partial sums, wherein the plurality of first partial sums correspond to results obtained by multiplying each input feature value from among the plurality of input feature values by a sign bit corresponding to each weight value from among the plurality of weight values; and obtaining an output feature value based on the inverted first partial sums, additional bits, and remaining partial sums other than the inverted first partial sums.

Inventors: **KIM; Ho Young (Suwon-si, KR)**

Applicant: **SAMSUNG ELECTRONICS CO., LTD.** (Suwon-si, KR)

Family ID: **93705189**

Assignee: **SAMSUNG ELECTRONICS CO., LTD. (Suwon-si, KR)**

Appl. No.: **18/896340**

Filed: **September 25, 2024**

Foreign Application Priority Data

KR	10-2024-0019510	Feb. 08, 2024

Publication Classification

Int. Cl.: **G06F17/15** (20060101); **G06N3/0464** (20230101)

U.S. Cl.:

Background/Summary

CROSS-REFERENCE TO RELATED APPLICATION

[0001] This application claims priority under 35 U.S.C. § 119 to Korean Patent Application No. 10-2024-0019510, filed on Feb. 8, 2024, in the Korean Intellectual Property Office, the disclosure of which is incorporated by reference herein in its entirety.

BACKGROUND

1. Field

[0002] The present disclosure relates to a convolution processing method and an electronic apparatus performing the convolution processing method.

2. Description of Related Art

[0003] A convolution neural network (CNN) may involve a significant amount of computations, so enhancing the energy efficiency of operations may be important. While some implementation may separate multiplication and addition for simplicity, this may not result in optimal energy efficiency.

SUMMARY

[0004] One or more embodiments may address at least the problems and/or disadvantages described above, and other disadvantages not described above. Also, the example embodiments are not required to overcome and may not overcome any of the problems and disadvantages described above.

[0005] In accordance with an aspect of the disclosure, a convolution processing method performed by at least one processor included in an electronic apparatus, includes: obtaining a plurality of partial sums by multiplying a plurality of input feature values in a binary form by a plurality of weight values in the binary form; inverting a plurality of first partial sums from among the plurality of partial sums, wherein the plurality of first partial sums correspond to results obtained by multiplying each input feature value from among the plurality of input feature values by a sign bit corresponding to each weight value from among the plurality of weight values; and obtaining an output feature value based on the inverted first partial sums, additional bits, and remaining partial sums other than the inverted first partial sums.

[0006] The convolution processing method may further include: predicting a bitwidth of a result of a convolution operation between the plurality of input feature values and the plurality of weight values; expanding the plurality of first partial sums by adding a plurality of first sequences to the plurality of first partial sums such that a length of each expanded first partial sum from among the plurality of expanded first partial sums matches the predicted bitwidth; converting the plurality of first sequences into a plurality of second sequences and summing a plurality of ones (“1s”); and determining the additional bits based on the plurality of second sequences and the summed plurality of ones (“1s”).

[0007] The additional bits may include a sum of the second sequences and a sum of the summed plurality of ones (“1s”).

[0008] The output feature value may be obtained by summing the remaining partial sums, the inverted first partial sums, and the additional bits.

[0009] The output feature value may be obtained using a compressor tree circuit.

[0010] The remaining partial sums, the inverted first partial sums, and the additional bits may be included in an input of a first stage of the compressor tree circuit.

[0011] The method may further include: based on a bias being present, sign-extending the bias, wherein the output feature value may be obtained by summing, using the compressor tree circuit, the sign-extended bias, the remaining partial sums, the inverted first partial sums, and the

additional bits.
[0012] The sign-extended bias, the remaining partial sums, the inverted first partial sums, and the additional bits may be included in a first stage of the compressor tree circuit.
[0013] The compressor tree circuit may include a plurality of first compressors and a plurality of second compressors, wherein each first compressor from among the plurality of first compressors may be configured to generate two outputs based on three inputs, and wherein each second compressor from among the plurality of second compressors may be configured to generate two outputs based on two inputs.
[0014] A remainder may be obtained by dividing a number of bits corresponding to a first place of a first stage, wherein based on the remainder being not equal to two (“2”), a number of the plurality of first compressors equal to a quotient obtained by dividing the number of bits by three (“3”) may be used for the first place of the first stage, and based on the remainder being equal to two (“2”), at least one first compressor or at least one second compressor may be used.
[0015] Based on at least one from among a bias and a residual input being present in a convolution operation, the at least one from among the bias and the residual input may be processed by the compressor tree circuit.
[0016] In accordance with an aspect of the disclosure, an electronic apparatus includes: a memory; and [0017] at least one processor operatively connected to the memory and configured to: obtain a plurality of partial sums by multiplying a plurality of input feature values in a binary form by a plurality of weight values in the binary form, invert a plurality of first partial sums from among the obtained plurality of partial sums, wherein the a plurality of first partial sums correspond to results obtained by multiplying each input feature value from among the plurality of input feature values by a sign bit corresponding to each weight from among the plurality of the weight values, and obtain an output feature value based on the inverted first partial sums, additional bits, and remaining partial sums other than the inverted first partial sums.
[0018] The at least one processor may be further configured to: predict a bitwidth of a result of a convolution operation between the plurality of input feature values and the plurality of weight values; expand the plurality of first partial sums by adding a plurality of first sequences to the plurality of first partial sums such that a length of each expanded first partial sum from among the plurality of expanded first partial sums matches the predicted bitwidth; convert the plurality of first sequences into a plurality of second sequences and sum a plurality of ones (“1s”); and [0019] determine the additional bits based on the plurality of second sequences and the summed plurality of ones (“1s”).
[0020] The additional bits may include a sum of the second sequences and a sum of the summed plurality of ones (“1s”).
[0021] The output feature value may be obtained by summing the remaining partial sums, the inverted first partial sums, and the additional bits.
[0022] The at least one processor may include a compressor tree circuit.
[0023] The remaining partial sums, the inverted first partial sums, and the additional bits may be included in an input of a first stage of the compressor tree circuit.
[0024] The at least one processor may be further configured to, based on a bias being present, sign-extend the bias and obtain the output feature value by summing, using the compressor tree circuit, the sign-extended bias, the remaining partial sums, the inverted first partial sums, and the additional bits.
[0025] The sign-extended bias, the remaining partial sums, the inverted first partial sums, and the additional bits may be included in a first stage of the compressor tree circuit.
[0026] A remainder may be obtained by dividing a number of bits corresponding to a first place of a first stage, the compressor tree circuit may include a plurality of first compressors and a plurality of second compressors, based on the remainder being not equal to two (“2”), a number of the plurality of first compressors equal to a quotient obtained by dividing the number of bits by three

(“3”) may be used for the first place of the first stage, and based on the remainder being equal to two (“2”), at least one first compressor or at least one second compressor may be used.

Description

BRIEF DESCRIPTION OF THE DRAWINGS

[0027] The above and other aspects, features, and advantages of certain embodiments of the present disclosure will be more apparent from the following description taken in conjunction with the accompanying drawings, in which:

[0028] FIG. **1**A is a diagram illustrating a deep learning operation method using an artificial neural network (ANN), according to an embodiment;

[0029] FIG. **1**B is a diagram illustrating a process of performing convolution processing (or a convolution operation) based on deep learning, according to an embodiment;

[0030] FIGS. **2** to **6** are diagrams illustrating convolution processing according to an embodiment;

[0031] FIGS. **7** to **9** are diagrams illustrating a compressor tree circuit according to an embodiment;

[0032] FIGS. **10** to **11** are diagrams illustrating an example of a method of configuring a compressor tree circuit according to an embodiment;

[0033] FIG. **12** is a diagram illustrating an example of an operation of an electronic apparatus when a bias or residual connection of convolution processing is present, according to an embodiment;

[0034] FIGS. **13**A and **13**B are block diagrams illustrating an electronic apparatus for performing convolution processing, according to an embodiment; and

[0035] FIG. **14** is a flowchart illustrating a convolution processing method of an electronic apparatus, according to an embodiment.

DETAILED DESCRIPTION

[0036] The following detailed structural or functional description is provided as an example only and various alterations and modifications may be made to embodiments without departing from the scope of the disclosure. Accordingly, the particular embodiments described herein are not intended to be limiting, and should be understood to include all changes, equivalents, and replacements within the idea and the scope of the disclosure.

[0037] Although terms, such as first, second, and the like are used to describe various components, the components are not limited to the terms. These terms should be used only to distinguish one component from another component. For example, a first component may be referred to as a second component, and similarly the second component may also be referred to as the first component.

[0038] It should be noted that if one component is described as being “connected”, “coupled”, or “joined” to another component, a third component may be “connected”, “coupled”, and “joined” between the first and second components, or the first component may be directly connected, coupled, or joined to the second component.

[0039] The singular forms “a”, “an”, and “the” are intended to include the plural forms as well, unless the context clearly indicates otherwise. It will be further understood that the terms “comprises/comprising” and/or “includes/including” when used herein, specify the presence of stated features, integers, steps, operations, elements, and/or components, but do not preclude the presence or addition of one or more other features, integers, steps, operations, elements, components and/or groups thereof.

[0040] Unless otherwise defined, all terms, including technical and scientific terms, used herein may have the same meaning as commonly understood by one of ordinary skill in the art to which this disclosure pertains. Terms, such as those defined in commonly used dictionaries, are to be interpreted as having a meaning that is consistent with their meaning in the context of the relevant art, and are not to be interpreted in an idealized or overly formal sense unless expressly so defined

herein.
[0041] Hereinafter, some embodiments are described in detail with reference to the accompanying drawings. When describing the embodiments with reference to the accompanying drawings, like reference numerals refer to like elements and a repeated, redundant, or duplicative description related thereto may be omitted.
[0042] FIG. **1**A is a diagram illustrating a deep learning operation method using an artificial neural network (ANN).
[0043] An artificial intelligence (AI) algorithm including deep learning or the like may involve inputting input data **101** to an ANN and learning output data **103** through an operation such as a convolution operation. The ANN may be a computational architecture that is modelled on a biological brain. In the ANN, nodes corresponding to neurons of a brain may be connected to each other and may collectively operate to process input data. Various types of neural networks may include, for example, a convolutional neural network (CNN), a recurrent neural network (RNN), a deep belief network (DBN), and a restricted Boltzmann machine (RBM), but embodiments are not limited thereto. In a feed-forward neural network, neurons may have links to other neurons. These links may extend in one direction, for example, a forward direction, through the neural network.
[0044] FIG. **1**A illustrates a structure in which the input data **101** is input to an ANN **102** (e.g., a CNN) and the output data **103** is output through the ANN **102**. The ANN **102** may be a deep neural network including two or more layers.
[0045] The CNN may be used to extract “features” such as a border and a line color from the input data **101**. The CNN may include a plurality of layers. Each of the layers may receive data, process data input to a corresponding layer, and generate data that is to be output from the corresponding layer. The data output from the layer may be a feature map generated by performing a convolution operation between an image or feature map input to the CNN and a weight value of one or more filters. Initial layers of the CNN may operate to extract low-level features such as edges or gradients from an input. Subsequent layers of the CNN may operate to extract gradually more complex features such as an eye and a nose in an image.
[0046] According to an embodiment, an electronic apparatus **100** may include a processor (or for example an operation circuit) configured to implement the ANN **102**. The electronic apparatus **100** may generate the output data **103** by processing the input data **101** using the processor.
[0047] FIG. **1**B is a diagram illustrating a process of performing convolution processing (which may be referred to as a convolution operation) based on deep learning.
[0048] Referring to FIG. **1**B, the process of performing convolution processing in the ANN **102** may be a process of generating an output feature map **130** by performing multiplication and addition operations between an input feature map **110** and a filter **120** in layers to generate output values, and accumulating and summing the output values.
[0049] The convolution processing process may refer to a process of performing multiplication and addition operations by applying a filter having a predetermined size, for example a filter **120** having a size of n×n (where n is a natural number), to the input feature map **110** from an upper left side or corner to a lower right side or corner in a current layer. In the example shown in FIG. **1**B, the process of performing a convolution operation using the filter **120** having a size of 3×3 is described. As described below, the filter **120** may correspond to 3×3×m (where m is a natural number). Here, m may represent the number of channels.
[0050] For example, first, an operation of multiplying 3×3 pieces of data in a first region **111** on the upper left side of the input feature map **110** by weights W.sub.11 to W.sub.33 of the filter **120**, respectively, may be performed. Here, the 3×3 pieces of data in the first region **111** may include a total of nine pieces of data (or nine input feature values X.sub.11 to X.sub.33) including three pieces of data in a first direction and three pieces of data in a second direction. Thereafter, first-first piece of output data (or a first-first output feature value Y.sub.11) in the output feature map **130** may be generated using a cumulative sum of the output values of the multiplication operation, for

example X.sub.11*W.sub.11, X.sub.12*W.sub.12, X.sub.13*W.sub.13, X.sub.21*W.sub.21, X.sub.22*W.sub.22, X.sub.23*W.sub.23, X.sub.31*W.sub.31, X.sub.32*W.sub.32, and X.sub.33*W.sub.33. Here, the symbol "*" may denote multiplication.

[0051] Thereafter, an operation may be performed by shifting the unit of data from the first region **111** to a second region **112** on the upper left side of the input feature map **110**. In this example, the number of pieces of data shifted in the input feature map **110** for the convolution processing process may be referred to as a stride. The size of the output feature map **130** to be generated may be determined based on the stride. For example, when the stride is equal to one ("1"), first-second piece of output data (or a first-second output feature value Y.sub.12) of the output feature map **130** may be generated by performing an operation of multiplying nine input feature values X.sub.12 to X.sub.34 included in the second region **112** by the weight values W.sub.11 to W.sub.33 of the filter **120** and accumulating and summing the output values of the multiplication operation, for example X.sub.12*W.sub.11, X.sub.13*W.sub.12, X.sub.14*W.sub.13, X.sub.22*W.sub.21, X.sub.23*W.sub.22, X.sub.24*W.sub.23, X.sub.32*W.sub.31, X.sub.33*W.sub.32, and X.sub.34*W.sub.33.

[0052] FIGS. **2** to **6** are diagrams illustrating convolution processing according to an embodiment.

[0053] FIG. **2** illustrates an example of a multiplication operation between an input feature value X.sub.11 **201** and a weight value W.sub.11 **202**. The input feature value X.sub.11 **201** may be a multiplicand of the multiplication operation. The weight value W.sub.11 **202** may be a multiplier corresponding to the multiplication operation.

[0054] Although FIGS. **2** to **6** illustrate examples in which a filter may have a size of 3×3×16, embodiments are not limited thereto, and in some embodiments the filter may have any size.

[0055] In the example shown in FIG. **2**, the input feature value X.sub.11 **201** may be an unsigned eight-bit number, and the weight value W.sub.11 **202** may be a signed eight-bit number. A most significant bit (MSB) b.sub.7 of the weight value W.sub.11 **202** may correspond to a sign bit. Although examples are described below in which the weight values may correspond to signed bits (e.g., signed eight-bit numbers, embodiments are not limited thereto, and the weight values may correspond to unsigned bits (e.g., unsigned eight-bit numbers). At least some or all of the description below may apply to embodiments in which the weight values include unsigned bits.

[0056] According to embodiments a bit value a.sub.0 and a bit value b.sub.0 may each be referred to as a bit in a 2.sup.0 place. For example, the bit value a.sub.0 may be referred to a bit in the 2.sup.0 place of the input feature value X.sub.11 **201**, and the bit value b.sub.0 may be referred to a bit in the 2.sup.0 place of the weight value W.sub.11 **202**. Similarly, a bit value a.sub.1 and a bit value b.sub.1 may each be referred to as a bit in a 2.sup.1 place, a bit value a.sub.2 and a bit value b.sub.2 may each be referred to as a bit in a 2.sup.2 place, and a bit value a.sub.3 and a bit value b.sub.3 may each be referred to as a bit in a 2.sup.3 place. In addition, a bit value a.sub.4 and a bit value b.sub.4 may each be referred to as a bit in a 2.sup.4 place, a bit value as and a bit value b.sub.5 may each be referred to as a bit in a 2.sup.5 place, a bit value a.sub.6 and a bit value b.sub.6 may each be referred to as a bit in a 2.sup.6 place, and a bit value a.sub.7 and a bit value b.sub.7 may each be referred to as a bit in a 2.sup.7 place.

[0057] The electronic apparatus **100** may generate a plurality of multiplication results by multiplying the input feature value X.sub.11 **201** by the weight value W.sub.11 **202**. For example, as shown in FIG. **2**, the plurality of multiplication results may include eight multiplication results, each of which is illustrated as a row of solid black circles in FIG. **2**. The sum of the plurality of multiplication results may correspond to X.sub.11*W.sub.11, so each multiplication result from among the plurality of multiplication results may be referred to as a partial sum. For example, as shown in FIG. **2**, the plurality of multiplication results may include a partial sum **210**, a partial sum **220**, a partial sum **230**, a partial sum **240**, a partial sum **250**, a partial sum **260**, a partial sum **270**, and a partial sum **280**.

[0058] According to embodiments, a partial sum **210** may correspond to a result obtained by

multiplying the input feature value X.sub.11 **201** by the bit b.sub.0 of the weight value W.sub.11 **202**, a partial sum **220** may correspond to the result obtained by multiplying the input feature value X.sub.11 **201** by the bit b.sub.1 of the weight value W.sub.11 **202**, a partial sum **230** may correspond to the result obtained by multiplying the input feature value X.sub.11 **201** by the bit b.sub.2 of the weight value W.sub.11 **202**, and a partial sum **240** may correspond to the result obtained by multiplying the input feature value X.sub.11 **201** by the bit b.sub.3 of the weight value W.sub.11 **202**. Similarly, a partial sum **250** may correspond to the result obtained by multiplying the input feature value X.sub.11 **201** by the bit b.sub.4 of the weight value W.sub.11 **202**, a partial sum **260** may correspond to the result obtained by multiplying the input feature value X.sub.11 **201** by the bit b.sub.5 of the weight value W.sub.11 **202**, a partial sum **270** may correspond to the result obtained by multiplying the input feature value X.sub.11 **201** by the bit b.sub.6 of the weight value W.sub.11 **202**, and a partial sum **280** may correspond to the result obtained by multiplying the input feature value X.sub.11 **201** by the bit by of the weight value W.sub.11 **202**.

[0059] The electronic apparatus **100** may predict the bitwidth of the result of a convolution operation between the input feature values X.sub.11 to X.sub.33 and the weight values of the 3×3×16 filter. For example, an input feature value may be represented by an unsigned eight-bit number, and a weight value may be represented by a signed eight-bit number. The result obtained by multiplying a single input feature value by a single weight value according to this example may correspond to a maximum of sixteen bits. The sum of 144 (e.g., 3×3×16) multiplication results may correspond to the result of the convolution operation between the input feature values X.sub.11 to X.sub.33 and the weight values of the 3×3×16 filter. Accordingly, the result of the convolution operation between the input feature values X.sub.11 to X.sub.33 and the weight values of the 3×3×16 filter may be about twenty-four bits. The electronic apparatus **100** may predict that the bitwidth of the result of the convolution operation between the input feature values X.sub.11 to X.sub.33 and the weight values of the 3×3×16 filter is twenty-four bits. According to an implementation, the electronic apparatus **100** may predict that the bitwidth of the result of the convolution operation is twenty-four bits or more (e.g., twenty-seven bits, etc.) by considering a bias and the like.

[0060] The electronic apparatus **100** may add a first sequence to the partial sum **280** (e.g., the result obtained by multiplying the input feature value X.sub.11 **201** by a sign bit of the weight value W.sub.11 **202**) to expand the partial sum **280** in order to generate an expanded partial sum **320**. For example, the electronic apparatus **100** may expand the partial sum **280** such that the length of the expanded partial sum matches a predicted bitwidth. For example, in the electronic apparatus **100**, the MSB of the partial sum **280** may be in the 2.sup.14 place, so the bitwidth of the partial sum **280** may be fifteen bits. The electronic apparatus **100** may add the first sequence including a plurality (e.g., nine) of zeroes (“0s”) to the partial sum **280** such that the bitwidth of the expanded partial sum **280** is greater than or equal to the predicted bitwidth (twenty-four bits). FIG. **3** illustrates an example in which the electronic apparatus **100** expands the partial sum **280** by adding a first sequence **310** including twelve zeros (“0s”) to the partial sum **280**, for ease of description.

[0061] The electronic apparatus **100** may invert an expanded partial sum **320**, which is determined by adding the first sequence **310** to the partial sum **280**. Inversion may include, for example, a NOT operation. When a zero (“0”) is inverted (or a NOT operation is performed on a value zero (“0’)), the zero (“0”) may be converted to a one (“1”). When a one (“1”) is inverted (or a NOT operation is performed on a one (“1”)), the one (“1”) may be converted to a zero (“0”). In the example shown in FIG. **4**, the symbol “-” may represent an inversion symbol (or, for example, a NOT operation symbol).

[0062] FIG. **4** illustrates an inverted first sequence **410** and an inverted partial sum **421**. The inverted first sequence **410** may correspond to a sequence including 12 ones (“1s”, and the inverted partial sum **421** may correspond to a result of inverting the partial sum **280**. The electronic apparatus **100** may invert the expanded partial sum **320** and add a one (“1”) **430** to the resulting

inverted sum. The one (“1”) **430** may correspond to the value in the 2.sup.7 place. The process in which the electronic apparatus **100** may invert the expanded partial sum **320** and adds one (“1”) **430** to the expanded partial sum **320**, which is inverted, may be similar to or substantially the same as the process in which the electronic apparatus **100** obtains a two's complement of the expanded partial sum **320**. For example, when an input feature value is A and a sign bit is one (“1”), the product of the input feature value and the sign bit may be −A. The electronic apparatus **100** may calculate the two's complement of A to represent-A. The two's complement of A may be calculated by calculating the one's complement of A and adding one (“1”) to the one's complement of A. The one (“1”) **430** described above may correspond to the one (“1”) that is added in the process of obtaining the two's complement of the expanded partial sum **320**. When the sign bit is zero (“0”), the product of the input feature value and the sign bit may be zero (“0”). Because the two's complement of zero (“0”) is zero (“0”), when the sign bit is zero (“0”), the electronic apparatus **100** may invert the result (e.g., zero (“0”)) of multiplying the input feature value by the sign bit, add one (“1”) to the inverted result, and perform operations described below.

[0063] In the example shown in FIG. **4**, partial sums **420** may include the inverted partial sum **421** and the plurality of partial sums (e.g., the partial sum **210** to the partial sum **270**) described with reference to FIG. **2**.

[0064] The description of the operation, provided with reference to FIGS. **2** to **4**, in which the electronic apparatus **100** may process (e.g., expand, invert, and add one (“1”)) the partial sum **280** corresponding to the result obtained by multiplying the input feature value X.sub.11 by the sign bit of the weight value W.sub.11 may apply to an operation in which the electronic apparatus **100** processes a partial sum corresponding to the result obtained by multiplying another input feature value by the sign bit of another weight value.

[0065] The electronic apparatus **100** may generate 144 ones (“1s”) corresponding to the 2.sup.7 place and generate 144 sequences (each of which may be referred to as a “second sequence”), each including a plurality (e.g., twelve) of ones (“1s”) by processing a partial sum corresponding to the result obtained by multiplying each input feature value by the sign bit of each weight value.

[0066] The electronic apparatus **100** may sum the generated ones (“1s”) and sum the generated second sequences. In the example shown in FIG. **5**, the electronic apparatus **100** may sum second sequences **510-1** to **510-***a* and sum 1s **530-1** to **530-***a*. As described with reference to FIGS. **2** to **4**, when the 3×3×16 filter is used, the electronic apparatus **100** may sum 144 second sequences **510-1** to **510-***a* and sum 144 ones (“1s”) **530-1** to **530-***a*. The sum of the second sequences **510-1** to **510-***a* and the sum of 1s **530-1** to **530-***a* are illustrated in FIG. **6**.

[0067] In the example shown in FIG. **6**, bits **610** (e.g., “1 1 1 1 0 1 1 1 0 0 0 0”) may represent the sum of the second sequences **510-1** to **510-***a* and bits **630** (e.g., “1 0 0 1”) may represent the sum of the ones (“1s”) **530-1** to **530-***a*. In the bits **610**, the bit value of each of the 2.sup.19 to 2.sup.21 and 2.sup.23 to 2.sup.26 places may correspond to a one (“1”). In the bits **630**, the bit value of each of the 2.sup.11 place and the 2.sup.14 place may correspond to a one (“1”).

[0068] The electronic apparatus **100** may sum the partial sums **520-1** to **520-***a* (which may, for example, correspond to the partial sums **420**) and the bits **610** and the bits **630**. The bits **610** and the bits **630** may correspond to bits for processing sign extension, for example. As described above, each of the partial sums **520-1** to **520-***a* may include a result of inverting a partial sum corresponding to a result obtained by multiplying each input feature value by a sign bit of each weight value.

[0069] The electronic apparatus **100** may sum the partial sums **520-1** to **520-***a* and the bits **610** and **630** using a compressor tree circuit. According to an embodiment, the compressor tree circuit may include a plurality of first compressors and a plurality of second compressors. A first compressor (e.g., a 3-to-2 compressor) may include a full adder and may generate two outputs (e.g., sum and carry-out bits) based on three inputs (e.g., two bits and a carry-in bit). A second compressor (e.g., a 2-to-2 compressor) may include a half adder and may generate two outputs based on two inputs

(e.g., two bits).
[0070] An example of a compressor tree circuit according to an embodiment is described with reference to FIGS. **7** to **9**.
[0071] FIGS. **7** to **9** are diagrams illustrating the compressor tree circuit according to an embodiment.
[0072] The compressor tree circuit may include a plurality of stages. The stages of the compressor tree circuit may include one or more first compressors. The last stage (e.g., stage 17 in FIG. **7**) of the compressor tree circuit may include a multi-bit adder. Some of the stages of the compressor tree circuit may include one or more second compressors in addition to one or more first compressors.
[0073] The compressor tree circuit may receive, at stage 1, the partial sums **520**-**1** to **520**-*a* and the bits **610** and **630** (hereinafter, referred to as "input data") described with reference to FIG. **6**. For example, as recorded in the cell corresponding to stage 1 and place 0 in a table **700** of FIG. **7**, the compressor tree circuit may receive, at stage 1, 144 bits corresponding to the 2.sup.0 place (e.g., place 0) of the input data. Compressors **800** corresponding to the 2.sup.0 place (e.g., place 0) of the compressor tree circuit are illustrated in FIG. **8**. In the example shown in FIG. **8**, a plurality of first compressors (e.g., 48 (=144/3) first compressors) at stage 1 may receive 144 bits corresponding to the 2.sup.0 place (e.g., place 0) of the input data. In FIG. **8**, the ones ("1s") place may correspond to the 2.sup.1 place. As recorded in the item of stage 1 and place 1 in the table **700** in FIG. **7**, the compressor tree circuit may receive, at stage 1, 288 bits corresponding to the 2.sup.1 place (e.g., place 1) of the input data. Examples of compressors **900** corresponding to the 2.sup.1 place (e.g., place 1) of the compressor tree circuit are illustrated in FIG. **9**. In the example shown in FIG. **9**, a plurality of first compressors (e.g., **96** (=288/3) first compressors) at stage 1 may receive 288 bits corresponding to the 2.sup.1 place (e.g., place 1) of the input data. In FIG. **9**, the zeros ("0s") place may represent the 2.sup.0 place and the twos ("2s") place may represent the 2.sup.2 place.
[0074] The compressor tree circuit may perform an operation at each stage. At stage 17, the compressor tree circuit may obtain a value (e.g., an output feature value) corresponding to the result of a convolution operation between input feature values and weight values by summing bits of places (e.g., 2.sup.0 place (e.g., place 0) to 2.sup.23 place (e.g., place 2.sup.3)) through the multi-bit adder.
[0075] According to embodiments, the items with bold borders in the table **700** of FIG. **7** may indicate that one or more second compressors are used. For example, the value two ("2") recorded in the cell corresponding to stage 5 and place 0 may indicate that one second compressor receives two bits. The value eight ("8") recorded in the cell corresponding to stage 8 and place 3 may indicate that four second compressors receive eight bits. The value five ("5") recorded in the cell corresponding to stage 11 and place 5 may indicate that two second compressors receive four out of five bits. In this case, the remaining one bit of the five bits may be transmitted to stage 12. The remaining cells other than the cells with bold borders and the cells with numbers less than 3 in the table **700** of FIG. **7** may indicate that a first compressor is used.
[0076] According to a comparative example, some implementations of a compressor tree circuit may include a Wallace tree circuit and an adder tree circuit. Table 1 compares examples of the compressor tree circuit including the Wallace tree circuit and the adder tree circuit with the compressor tree circuit according to an embodiment.
TABLE-US-00001 TABLE 1 Compressor tree circuit according Compressor tree to a comparative circuit according to example an embodiment Decline rate Number of flip- 5652 2333 58.7% flops Sum of the 11066 9193 16.9% number of full adders and the number of half adders Number of 6 5 16.7% pipeline stages
[0077] In contrast with the comparative example discussed above, the compressor tree circuit according to an embodiment may include fewer flip-flops and may include fewer adders. Additionally, the number of pipeline stages may be less in the compressor tree circuit according to an embodiment than in the compressor tree circuit according to the comparative example. In

general, a flip-flop may consume a relatively large amount of power, but the compressor tree circuit according to an embodiment may include a smaller number of flip-flops, thereby improving power efficiency. Furthermore, the compressor tree circuit according to an embodiment may include a smaller number of adders, thereby reducing an area.

[0078] Moreover, in terms of power per cell area, the compressor tree circuit according to the comparative example may have a total power of 80 milliwatts (mW) or more when the frequency is 300 megahertz (MHz), but the compressor tree circuit according to an embodiment may have a total power of less than 80 mW when the frequency is 300 MHz to 400 MHz. Accordingly, the compressor tree circuit according to an embodiment may have various mixable frequencies and may have relatively good power efficiency at the same frequency.

[0079] FIGS. **10** and **11** are diagrams illustrating examples of a method of configuring a compressor tree circuit, according to an embodiment.

[0080] In the example shown in FIG. **10**, the bitwidth of the result of a convolution operation between input feature values and weight values of a 3×3×16 filter may be, for example, twenty-five.

[0081] The number of bits in the 2.sup.0 place (e.g., place 0) to 2.sup.14 place (e.g., place 14) may be 144, 288, 432, 576, 720, 864, 1008, 1152, 1008, 864, 720, 576, 432, 288, and 144, respectively, as shown in the row corresponding to stage 1 in FIG. **10**.

[0082] At each stage, the number of first compressors for the bits in each place may correspond to the quotient obtained by dividing the number of bits of each place by three (“3”). For example, at stage 1, there may be 144 bits in the 2.sup.0 place (e.g., place 0), and the number of first compressors for the bits in the 2.sup.0 place (e.g., place 0) may correspond to the quotient (e.g., forty-eight) obtained by dividing 144 by three. A first compressor may produce two outputs based on three inputs, with one (e.g. a sum) of the two outputs being transmitted to the next stage and the other (e.g. a carry-out bit) of the two outputs being transmitted to the next place (e.g., a 2.sup.1 place). The forty-eight first compressors may output ninety-six bits. Of the ninety-six bits, forty-eight bits may be transmitted to stage 2 and the remaining forty-eight bits may be transmitted to the 2.sup.1 place (e.g., place 1) of stage 1. At stage 2, there may be forty-eight bits in the 2.sup.0 place (e.g., place 0). The number of first compressors for the bits in the 2.sup.0 place (e.g., place 0) may correspond to the quotient (e.g., sixteen) obtained by dividing forty-eight by three. At stage 3, there may be sixteen bits in the 2.sup.0 place (e.g., place 0). The number of first compressors for the bits in the 2.sup.0 place (e.g., place 0) may correspond to the quotient (e.g., five) obtained by dividing sixteen by three. At stage 4, there may be six bits in the 2.sup.0 place (e.g., place 0). The number of first compressors for the bits in the 2.sup.0 place (e.g., place 0) may correspond to the quotient (e.g., two) obtained by dividing six by three. Similarly, at each stage, the number of first compressors for the bits in each place may be determined.

[0083] In the example shown in FIG. **10**, among the cells in a table **1000**, the shaded cells may have a remainder of two (“2”) when the number of bits is divided by three. For example, at stage 5, the number of bits in the 2.sup.1 place (e.g., place 1) may be eleven. When eleven is divided by three, the remainder may be two.

[0084] In some embodiments first compressor or a second compressor may be used for the shaded items. Accordingly, the total number of first compressors and the total number of second compressors may be combined in various ways. For example, there may be a combination of m first compressors and n second compressors and there may be a combination of m first compressors and n+1 second compressors. There may be conditions for selecting one of various combinations.

[0085] For example, a first condition may be a condition for selecting a combination with the maximum number of first compressors, among various combinations. A second condition may be a condition for selecting a combination with the smallest number of second compressors when various combinations have the same number of first compressors. A third condition may be a condition for selecting a combination in which a second compressor is at a later stage, when

various combinations have the same number of first compressors and the same number of second compressors. For example, combination A may have m first compressors and n second compressors and combination B may have m first compressors and n second compressors. The number of first compressors in combination A and the number of first compressors in combination B may be the same. The number of second compressors in combination A and the number of second compressors in combination B may also be the same. For combination A, a second compressor may be used at stage 13. For combination B, a second compressor may be used at stage 14. In this case, combination B may be selected.

[0086] An example of a combination selected according to the first condition, second condition, or third condition is shown in a table **1100** of FIG. **11**. A second compressor may be used for each of the cells with bold borders in the table **1100**. For example, the value "5" recorded in the cell corresponding to stage 9 and place 3 may indicate that two second compressors are used.

[0087] FIG. **12** is a diagram illustrating an example of an operation of an electronic apparatus when a bias or a residual connection of convolution processing is present.

[0088] According to an embodiment, a convolution processing process (or a convolution operation process) may have a bias or a residual connection. In this case, at stage 1, the number of bits in each place may increase by one. For example, compared to the example described with reference to FIG. **7**, in the example shown in FIG. **12**, the number of bits in each place at stage 1 may increase by one. More particularly, the electronic apparatus **100** may sign-extend the bias, and the sign-extended bias and the partial sums **520-1** to **520**-*a* and bits **610** and **630** described with reference to FIG. **6** may be input into a compressor tree circuit. In some embodiments, sign-extending may refer to increasing the number of bits of a binary number while preserving a sign (e.g., positive/negative) and a value of the binary number, however embodiments are not limited thereto. When the residual connection is present, bits input through the residual connection may be input into the compressor tree circuit along with the partial sums **520-1** to **520**-*a* and the bits **610** and **630**.

[0089] According to an embodiment, compared to the example described with reference to FIG. **7**, the number of stages may not change in the example shown in FIG. **12**, even when a bias or a residual connection may be present. Accordingly, latency may not increase even when a bias or a residual connection may be present.

[0090] FIGS. **13**A and **13**B are block diagrams illustrating an electronic apparatus for performing convolution processing, according to an embodiment.

[0091] An electronic apparatus **1300** (which may correspond, for example, to the electronic apparatus **100**) may include various computing devices, such as a mobile phone, a smartphone, a tablet personal computer (PC), an e-book device, a laptop, a PC, a desktop, a workstation, or a server, various wearable devices, such as a smart watch, smart eyeglasses, a head-mounted display (HMD), or smart clothing, various home appliances such as a smart speaker, a smart television (TV), or a smart refrigerator, and other devices, such as a smart car, a smart kiosk, an Internet of things (IoT) device, a walking assist device (WAD), a drone, or a robot.

[0092] Referring to FIG. **13**A, the electronic apparatus **1300** may include a processor **1310** and a memory **1320**.

[0093] The processor **1310** may be, for example, a hardware accelerator and may correspond to a neural processing unit (NPU), a graphics processing unit (GPU), and the like. In some embodiments, the electronic apparatus **1300** may include a host processor (e.g., a central processing unit (CPU)) that may generally control the electronic apparatus **1300**. The host processor may receive, from the processor **1310**, the execution result of the processor **1310**. The host processor may display the received execution result on a display and the like of the electronic apparatus **1300**.

[0094] The memory **1320** may include, for example, an L2 memory. The memory **1320** may store, for example, an input feature map and a filter. The processor **1310** may receive the input feature map and the filter from the memory **1320** and may perform convolution processing (or a

convolution operation) based on the input feature map and the filter.
[0095] According to an embodiment, the processor **1310** may include a multiply and accumulation (MAC) circuit **1311**, a memory **1312**, and an activation circuit **1313**, as illustrated in FIG. **13**B. The memory **1312** may include, for example, static random access memory (SRAM). The processor **1310** may store the received input feature map and the filter in the memory **1312**. The activation circuit **1313** may correspond to a circuit that implements an activation function (or a rectified linear unit (ReLU)) of a CNN.
[0096] According to an embodiment, the MAC circuit **1311** may include a compressor tree circuit according to an embodiment. The MAC circuit **1311** may receive input feature values and weight values of a filter from the memory **1312**. The MAC circuit **1311** may perform an operation based on the input feature values and the weight values of the filter.
[0097] According to an embodiment, the processor **1310** (or the MAC circuit **1311**) may obtain a plurality of partial sums by multiplying input feature values in a binary form (e.g., the input feature values X.sub.11 to X.sub.33 in the binary form) respectively by weight values in a binary form (e.g., the weight values of the 3×3×16 filter). Each of the input feature values may correspond to, for example, an unsigned binary number. Each of the weight values may correspond to, for example, a signed binary number.
[0098] The processor **1310** may invert first partial sums of the obtained partial sums, the first partial sums corresponding to results obtained by multiplying each of the input feature values by the sign bit of each of the weight values. For example, the processor **1310** may invert first partial sums (e.g., the partial sum **280** of FIG. **3**) corresponding to the result obtained by multiplying the input feature value X.sub.11 and the sign bit of the weight value W.sub.11.
[0099] The processor **1310** (or the MAC circuit **1311**) may obtain an output feature value based on the remaining partial sums (e.g., the remaining partial sums other than the inverted partial sums among the partial sums **520-1** to **520-***a* in FIG. **6**) other than the inverted first partial sums, the inverted first partial sums (e.g., the inverted partial sums in FIG. **6**), and bits (e.g., the bits **610** and the bits **630** of FIG. **6**). For example, the processor **1310** (or the MAC circuit **1311**) may obtain the output feature value by summing the remaining partial sums, the inverted first partial sums, and the bits **610** and the bits **630**. In other words, the processor **1310** (or the MAC circuit **1311**) may obtain the output feature value by summing the partial sums **520-1** to **520-***a* and the bits **610** and the bits **630** described with reference to FIG. **6**. The obtained output feature value may correspond to the result of a convolution operation between the input feature values and the weight values.
[0100] According to an embodiment, the remaining partial sums other than the inverted first partial sums, the inverted first partial sums, and the bits **610** and the bits **630** may correspond to an input of a first stage (e.g., stage 1 described with reference to FIG. **7**) of the MAC circuit **1311** (or a compressor tree circuit).
[0101] According to an embodiment, the processor **1310** may predict the bitwidth of the result of the convolution operation between the input feature values and the weight values. The processor **1310** may expand the first partial sums by adding first sequences (e.g., the first sequence **310** of FIG. **3**) respectively to the first partial sums such that the length of each of the expanded first partial sums matches the predicted bitwidth. The processor **1310** may convert the added first sequences respectively into second sequences and add a plurality of 1s (e.g., the 1s **530-1** to **530-***a* of FIG. **5**). The processor **1310** may determine the bits **610** and **630** described above based on the second sequences (e.g., the second sequences **510-1** to **510-***a* of FIG. **5**) and the added ones (“1s”). For example, the processor **1310** may determine the bits **610** and **630** including the sum (e.g., **610**) of the second sequences and the sum (e.g., **630**) of the added ones (“1s”).
[0102] According to an embodiment, when a bias is present, the processor **1310** may sign-extend the bias. The processor **1310** may obtain the output feature value by summing, through the MAC circuit **1311** (or the compressor tree circuit), the sign-extended bias, the remaining partial sums other than the inverted first partial sums, the inverted first partial sums, and the bits **610** and **630**. In

this case, the sign-extended bias, the remaining partial sums, the inverted first partial sums, and the bits **610** and **630** may correspond to an input of a first stage (e.g., stage 1 described with reference to FIG. **12**) of the compressor tree circuit.

[0103] According to an embodiment, the MAC circuit **1311** (or the compressor tree circuit) may include a plurality of first compressors and a plurality of second compressors. Each of the first compressors may generate two outputs based on three inputs. Each of the second compressors may generate two outputs based on two inputs.

[0104] According to an embodiment, a remained may be obtained by dividing the number of bits corresponding to the first place of a first stage. Based on the remainder being not equal to two ("2"), a number of first compressors equal to the quotient obtained by dividing the number of bits by three may be used for the first place of the first stage. When the remainder is equal to two ("2"), at least one first compressor or at least one second compressor may be optionally used. For example, in the example shown in FIG. **7**, the number of bits corresponding to the 2.sup.3 place (e.g., place 3) of stage 4 may be 106. When 106 is divided by three, the remainder may be one ("1") and the quotient may be thirty-five ("35"). In this case, thirty-five first compressors may be used for the 2.sup.3 place (e.g., place 3) of stage 4. In the example shown in FIG. **7**, the number of bits corresponding to the 2.sup.1 place (e.g., place 1) of stage 5 may be eleven. When eleven is divided by three, the remainder may be two ("2") and the quotient may be three ("3"). In this case, three first compressors may be used for the 2.sup.1 place (e.g., place 1) of stage 5. In some embodiments, when eleven is divided by two, the remainder may be one ("1") and the quotient may be five ("5"), and five second compressors may be used for the 2.sup.1 place (e.g., place 1) of stage 5. For optimization purposes, three first compressors may be used rather than five second compressors for the 2.sup.1 place (e.g., place 1) of stage 5.

[0105] According to an implementation, when the remainder is equal to two ("2"), first and second compressors may not be used.

[0106] According to an embodiment, when a bias or a residual input (e.g., an input through a residual connection) is present in a convolution operation, the MAC circuit **1311** may process the bias or the residual input.

[0107] According to an embodiment, even when a bias or a residual input is present in the convolution operation, the MAC circuit **1311** (or the compressor tree circuit) may process the bias or the residual input without increasing the number of stages. For example, when comparing the example described with reference to FIG. **12** with the example described with reference to FIG. **7**, the total number of stages may be the same at seventeen, so when a bias is present, the bias may be processed without increasing the number of stages.

[0108] The operation of the electronic apparatus **100** described with reference to FIGS. **1** to **12** may apply to the electronic apparatus **1300** described with reference to FIG. **13**.

[0109] FIG. **14** is a flowchart illustrating a convolution processing method of an electronic apparatus, according to an embodiment.

[0110] Referring to FIG. **14**, in operation **1410**, the electronic apparatus **1300** may obtain a plurality of partial sums by multiplying input feature values in the binary form respectively by weight values in a binary form.

[0111] In operation **1420**, the electronic apparatus **1300** may invert first partial sums from among the obtained partial sums, wherein the first partial sums correspond to the results obtained by multiplying each of the input feature values by the sign bit of each of the weight values.

[0112] In operation **1430**, the electronic apparatus **1300** may obtain an output feature value based on the remaining partial sums other than the inverted first partial sums, the inverted first partial sums, and the bits **610** and **630**. For example, the electronic apparatus **1300** may obtain the output feature value by summing the remaining partial sums, the inverted first partial sums, and the bits **610** and the bits **630**. In this case, the remaining partial sums, the inverted first partial sums, and the bits **610** and the bits **630** may be included in an input of the first stage of a compressor tree circuit.

In embodiments, the bits **610** and the bits **630** may be referred to as additional bits.

[0113] According to an embodiment, the electronic apparatus **1300** may predict the bitwidth of the result of the convolution operation between the input feature values and the weight values. The electronic apparatus **1300** may expand the first partial sums by adding first sequences respectively to the first partial sums. In this case, the electronic apparatus **1300** may expand the first partial sums such that the length of each of the expanded first partial sums matches the predicted bitwidth. The electronic apparatus **1300** may convert the added first sequences respectively into second sequences and add a plurality of ones ("1s"). The electronic apparatus **1300** may determine the bits **610** and the bits **630** based on the second sequences and the added ones ("1s"). The bits **610** and the bits **630** may include, for example, the sum of the second sequences and the sum of the added ones ("1s").

[0114] According to an embodiment, when a bias is present, the electronic apparatus **1300** may sign-extend the bias. In operation **1430**, the electronic apparatus **1300** may obtain the output feature value by summing, through the compressor tree circuit, the sign-extended bias, the remaining partial sums, the inverted first partial sums, and the bits **610** and **630**. In this case, the sign-extended bias, the remaining partial sums, the inverted first partial sums, and the bits **610** and **630** may be included in the input of the first stage of the compressor tree circuit.

[0115] According to an embodiment, the electronic apparatus **1300** may multiply an unsigned multiplicand by a signed multiplier. In this case, the results (e.g., the partial sum **210** to partial sum **270** of FIG. **2**) obtained by multiplying a multiplicand by each of the bits other than a sign bit may be a positive. Therefore, the electronic apparatus **1300** may not sign-extend each of the results (e.g., the partial sum **210** to partial sum **270** of FIG. **2**). The result (e.g., the partial sum **270**) obtained by multiplying the multiplicand by the sign bit may be negative or zero ("0"). The electronic apparatus **1300** may perform a process of obtaining the two's complement of the result (e.g., the partial sum **270**) of multiplying the multiplicand by the sign bit, may not perform sign-extension, and may sum ones ("1s") (e.g., the ones ("1s") **530-1** to **530-***a*) in advance before summing partial sums (e.g., the partial sums **520-1** to **520-***a*). Accordingly, the electronic apparatus **1300** may reduce the number of operations of the convolution operation.

[0116] The description provided with reference to FIGS. **1** to **13** may apply to the convolution processing method described with reference to FIG. **14**.

[0117] The embodiments described herein may be implemented using a hardware component, a software component, and/or a combination thereof. A processing device may be implemented using one or more general-purpose or special-purpose computers, such as, for example, a processor, a controller and an arithmetic logic unit (ALU), a digital signal processor (DSP), a microcomputer, a field-programmable gate array (FPGA), a programmable logic unit (PLU), a microprocessor, or any other device capable of responding to and executing instructions in a defined manner. The processing device may run an operating system (OS) and one or more software applications that run on the OS. The processing device may also access, store, manipulate, process, and create data in response to execution of the software. For purpose of simplicity, the description of a processing device is singular; however, one of ordinary skill in the art will appreciate that a processing device may include multiple processing elements and multiple types of processing elements. For example, the processing device may include a plurality of processors, or a single processor and a single controller. In addition, different processing configurations are possible, such as parallel processors.

[0118] The software may include a computer program, a piece of code, an instruction, or one or more combinations thereof, to independently or collectively instruct or configure the processing device to operate as desired. Software and data may be embodied permanently or temporarily in any type of machine, component, physical or virtual equipment, computer storage medium or device, or in a propagated signal wave capable of providing instructions or data to or being interpreted by the processing device. The software may also be distributed over network-coupled computer systems so that the software is stored and executed in a distributed fashion. The software

and data may be stored by one or more non-transitory computer-readable recording mediums.
[0119] The methods according to the above-described embodiments may be recorded in non-transitory computer-readable media including program instructions to implement various operations of the above-described embodiments. The media may also include, alone or in combination with the program instructions, data files, data structures, and the like. The program instructions recorded on the media may be those specially designed and constructed for the purposes of embodiments, or they may be of the kind well-known and available to those having skill in the computer software arts. Examples of non-transitory computer-readable media include magnetic media such as hard disks, floppy disks, and magnetic tape; optical media such as CD-ROM discs and DVDs; magneto-optical media such as floptical disks; and hardware devices that are specially configured to store and perform program instructions, such as read-only memory (ROM), RAM, flash memory, and the like. Examples of program instructions include both machine code, such as produced by a compiler, and files containing higher-level code that may be executed by the computer using an interpreter.
[0120] The above-described hardware devices may be configured to act as one or more software modules in order to perform the operations of the above-described embodiments, or vice versa.
[0121] As described above, although the embodiments are described with reference to the limited drawings, one of ordinary skill in the art may apply various technical modifications and variations based thereon without departing from the scope of the disclosure. For example, suitable results may be achieved if the described techniques are performed in a different order, and/or if components in a described system, architecture, device, or circuit are combined in a different manner, or replaced or supplemented by other components or their equivalents.
[0122] Therefore, other implementations, other embodiments, and equivalents of the claims are within the scope of the following claims.

Claims

1. A convolution processing method performed by at least one processor included in an electronic apparatus, the convolution processing method comprising: obtaining a plurality of partial sums by multiplying a plurality of input feature values in a binary form by a plurality of weight values in the binary form; inverting a plurality of first partial sums from among the plurality of partial sums, wherein the plurality of first partial sums correspond to results obtained by multiplying each input feature value from among the plurality of input feature values by a sign bit corresponding to each weight value from among the plurality of weight values; and obtaining an output feature value based on the inverted first partial sums, additional bits, and, remaining partial sums other than the inverted first partial sums.
2. The convolution processing method of claim 1, further comprising: predicting a bitwidth of a result of a convolution operation between the plurality of input feature values and the plurality of weight values; expanding the plurality of first partial sums by adding a plurality of first sequences to the plurality of first partial sums such that a length of each expanded first partial sum from among the plurality of expanded first partial sums matches the predicted bitwidth; converting the plurality of first sequences into a plurality of second sequences and summing a plurality of ones (“1s”); and determining the additional bits based on the plurality of second sequences and the summed plurality of ones (“1s”).
3. The convolution processing method of claim 2, wherein the additional bits comprise a sum of the second sequences and a sum of the summed plurality of ones (“1s”).
4. The convolution processing method of claim 1, wherein the output feature value is obtained by summing the remaining partial sums, the inverted first partial sums, and the additional bits.
5. The convolution processing method of claim 1, wherein the output feature value is obtained using a compressor tree circuit.

6. The convolution processing method of claim 5, wherein the remaining partial sums, the inverted first partial sums, and the additional bits are included in an input of a first stage of the compressor tree circuit.

7. The convolution processing method of claim 5, further comprising: based on a bias being present, sign-extending the bias, wherein the output feature value is obtained by summing, using the compressor tree circuit, the sign-extended bias, the remaining partial sums, the inverted first partial sums, and the additional bits.

8. The convolution processing method of claim 7, wherein the sign-extended bias, the remaining partial sums, the inverted first partial sums, and the additional bits are included in a first stage of the compressor tree circuit.

9. The convolution processing method of claim 5, wherein the compressor tree circuit comprises a plurality of first compressors and a plurality of second compressors, wherein each first compressor from among the plurality of first compressors is configured to generate two outputs based on three inputs, and wherein each second compressor from among the plurality of second compressors is configured to generate two outputs based on two inputs.

10. The convolution processing method of claim 9, wherein a remainder is obtained by dividing a number of bits corresponding to a first place of a first stage, wherein based on the remainder being not equal to two (“2”), a number of the plurality of first compressors equal to a quotient obtained by dividing the number of bits by three (“3”) are used for the first place of the first stage, and based on the remainder being equal to two (“2”), at least one first compressor or at least one second compressor is used.

11. The convolution processing method of claim 5, wherein, based on at least one from among a bias and a residual input being present in a convolution operation, the at least one from among the bias and the residual input is processed by the compressor tree circuit.

12. An electronic apparatus comprising: a memory; and at least one processor operatively connected to the memory and configured to: obtain a plurality of partial sums by multiplying a plurality of input feature values in a binary form by a plurality of weight values in the binary form, invert a plurality of first partial sums from among the obtained plurality of partial sums, wherein the a plurality of first partial sums correspond to results obtained by multiplying each input feature value from among the plurality of input feature values by a sign bit corresponding to each weight from among the plurality of the weight values, and obtain an output feature value based on the inverted first partial sums, additional bits, and remaining partial sums other than the inverted first partial sums.

13. The electronic apparatus of claim 12, wherein the at least one processor is further configured to: predict a bitwidth of a result of a convolution operation between the plurality of input feature values and the plurality of weight values; expand the plurality of first partial sums by adding a plurality of first sequences to the plurality of first partial sums such that a length of each expanded first partial sum from among the plurality of expanded first partial sums matches the predicted bitwidth; convert the plurality of first sequences into a plurality of second sequences and sum a plurality of ones (“1s”); and determine the additional bits based on the plurality of second sequences and the summed plurality of ones (“1s”).

14. The electronic apparatus of claim 13, wherein the additional bits comprise a sum of the second sequences and a sum of the summed plurality of ones (“1s”).

15. The electronic apparatus of claim 12, wherein the output feature value is obtained by summing the remaining partial sums, the inverted first partial sums, and the additional bits.

16. The electronic apparatus of claim 12, wherein the at least one processor comprises a compressor tree circuit.

17. The electronic apparatus of claim 16, wherein the remaining partial sums, the inverted first partial sums, and the additional bits are included in an input of a first stage of the compressor tree circuit.

18. The electronic apparatus of claim 16, wherein the at least one processor is further configured to, based on a bias being present, sign-extend the bias and obtain the output feature value by summing, using the compressor tree circuit, the sign-extended bias, the remaining partial sums, the inverted first partial sums, and the additional bits.

19. The electronic apparatus of claim 18, wherein the sign-extended bias, the remaining partial sums, the inverted first partial sums, and the additional bits are included in a first stage of the compressor tree circuit.

20. The electronic apparatus of claim 16, wherein a remainder is obtained by dividing a number of bits corresponding to a first place of a first stage, wherein the compressor tree circuit comprises a plurality of first compressors and a plurality of second compressors, wherein based on the remainder being not equal to two (“2”), a number of the plurality of first compressors equal to a quotient obtained by dividing the number of bits by three (“3”) are used for the first place of the first stage, and wherein based on the remainder being equal to two (“2”), at least one first compressor or at least one second compressor is used.