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

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
27 January 2022 (27.01.2022)

WIPO | PCT

(10) International Publication Number
WO 2022/018424 A2

(51) International Patent Classification:
*G06N 3/04* (2006.01) *G06N 7/00* (2006.01)
*G06N 3/063* (2006.01) *G06N 3/08* (2006.01)

(21) International Application Number:
PCT/GB2021/051855

(22) International Filing Date:
19 July 2021 (19.07.2021)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:
2011510.1 24 July 2020 (24.07.2020) GB

(71) Applicant: **ARM LIMITED** [GB/GB]; 110 Fulbourn Road, Cambridge, Cambridgeshire CB1 9NJ (GB).

(72) Inventors: **VENU, Balaji**; c/o Arm Limited, 110 Fulbourn Road, Cambridge CB1 9NJ (GB). **EYOLE, Mbou**; c/o Arm Limited, 110 Fulbourn Road, Cambridge CB1 9NJ (GB).

(74) Agent: **TLIP LTD** et al.; Association No 587, 14 King Street, Leeds LS1 2HL (GB).

(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, DJ, DK, DM, DO, DZ, EC, EE, EG, ES, FI, GB, GD, GE, GH, GM, GT, HN, HR, HU, ID, IL, IN, IR, IS, IT, JO, JP, KE, KG, KH, KN, KP, KR, KW, 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, ST, SV, SY, TH, TJ, TM, TN, TR, TT, TZ, UA, UG, US, UZ, VC, VN, WS, ZA, ZM, ZW.

(54) Title: CERTAINTY-BASED CLASSIFICATION NETWORKS

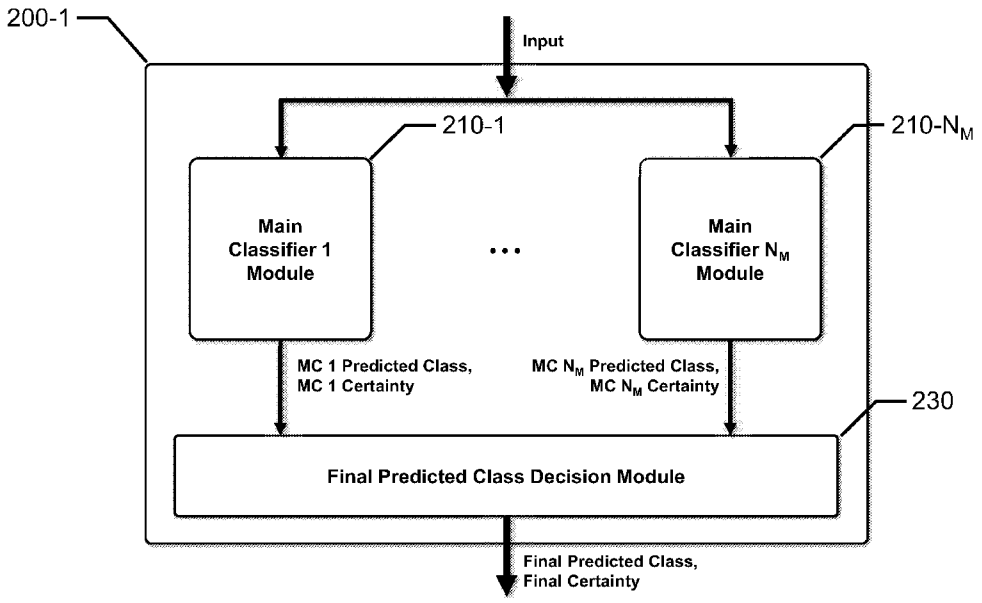


FIG. 4A

(57) Abstract: A certainty-based prediction apparatus and method are provided. A plurality of main classifier (MC) modules each predict an MC predicted class based on input data, and determine an MC certainty. Each MC module processes a pre-trained, machine learning main classifier having at least one expert class and a plurality of non-expert classes. An expert classifier (EC) module associated with each expert class predicts an EC predicted class based on the input data. Each EC module processes a pre-trained, machine learning expert classifier having two classes including an associated expert class and a residual class that includes any non-associated expert classes and the plurality of non-expert classes. A final predicted class decision module determines a final predicted class and a final certainty based on each MC predicted class, each MC certainty and each EC predicted class. The final predicted class and the final certainty are output.

WO 2022/018424 A2

*[Continued on next page]*

**(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, 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:**

— *without international search report and to be republished upon receipt of that report (Rule 48.2(g))*

# CERTAINTY-BASED CLASSIFICATION NETWORKS

## RELATED APPLICATION

[0001] The content of United Kingdom Patent Application No. GB2011510.1, filed on 24 July 2020, is incorporated herein by reference in its entirety.

## BACKGROUND

[0002] The present disclosure relates to computer systems. More particularly, the present disclosure relates to certainty-based classification networks.

[0003] Prediction is a fundamental element of many classification networks that include machine learning (ML), such as, for example, artificial neural networks (ANNs), convolutional neural networks (CNNs), recurrent neural networks (RNNs), Binary Neural Networks (BNN), Support Vector Machines (SVMs), Decision Trees, Bayesian networks, Naïve Bayes, etc. For example, safety-critical systems may implement classification networks for certain critical tasks, particularly in autonomous vehicles, robotic medical equipment, etc.

[0004] However, a classification network never achieves 100% prediction accuracy due to many reasons, such as, for example, insufficient data for a class, out of distribution (OOD) input data (i.e., data that do not belong to any of the classes), etc. Classification networks implemented in both hardware and software are also susceptible to hard and soft errors, which may worsen the prediction accuracy or lead to a fatal event. Generally, classification networks simply provide the “best” prediction based on the input data and the underlying training methodology and data.

[0005] Unfortunately, classification networks do not distinguish between correct and incorrect predictions, which can be fatal for many systems in general, and for safety-critical systems in particular.

## BRIEF DESCRIPTION OF THE DRAWINGS

[0006] FIG. 1 depicts an ANN, in accordance with embodiments of the present disclosure.

[0007] FIGS. 2A and 2B depict prediction accuracy for an ANN, in accordance with embodiments of the present disclosure.

[0008] FIG. 3 depicts a block diagram of a system, in accordance with embodiments of the present disclosure.

[0009] FIGS. 4A, 4B and 4C depict block diagrams of a hardware accelerator, in accordance with embodiments of the present disclosure.

[0010] FIG. 5 depicts a flow diagram presenting functionality for a certainty-based prediction process, in accordance with embodiments of the present disclosure.

[0011] FIG. 6A depicts a block diagram of a training system for a machine learning main classifier, in accordance with embodiments of the present disclosure.

[0012] FIG. 6B depicts a block diagram of a threshold determination process for a machine learning main classifier, in accordance with an embodiment of the present disclosure.

[0013] FIG. 6C depicts a block diagram of a training system for a machine learning expert classifier, in accordance with embodiments of the present disclosure.

DETAILED DESCRIPTION

[0014] Embodiments of the present disclosure will now be described with reference to the drawing figures, in which like reference numerals refer to like parts throughout.

[0015] Embodiments of the present disclosure advantageously provide an ensemble of classification networks that identify and reduce the number of incorrect predictions based on a level of confidence, or certainty, for each prediction. In many embodiments, a prediction may have a high level of confidence (i.e., a certain prediction) or a low level of confidence (i.e., an uncertain prediction); in other embodiments, a range of confidence levels may be provided. A certain prediction is processed normally, while an uncertain prediction is subject to additional processing

that may promote the uncertain prediction to a certain prediction, may replace the uncertain prediction, discard the uncertain prediction, etc.

[0016] Generally, certainty divides the number of correct predictions for a baseline classification network into a number of correct and certain predictions (e.g., "I know" this prediction is correct) and a number of correct and uncertain predictions (e.g., "I don't know" whether this prediction is correct). Similarly, certainty divides the number of incorrect predictions for the baseline classification network into a number of incorrect and certain predictions (e.g., "I know" that this prediction is incorrect) and a number of incorrect and uncertain predictions (e.g., "I don't know" whether this prediction is incorrect). While certainty reduces the number of correct predictions of the baseline classification network to a small degree by identifying the correct and uncertain predictions, certainty significantly reduces the number of incorrect predictions of the baseline classification network by identifying the incorrect and uncertain predictions, which is advantageous for many classification systems.

[0017] In one embodiment, a hardware accelerator includes a plurality of main classifier (MC) modules, an expert classifier (EC) module associated with each expert class, and a final predicted class decision module coupled to each MC module and each EC module. Each MC module processes a pre-trained, machine learning main classifier having at least one expert class and a plurality of non-expert classes, and each MC module is configured to predict an MC predicted class based on input data, determine an MC certainty, and output the MC predicted class and the MC certainty. Each EC module processes a pre-trained, machine learning expert classifier having two classes including an associated expert class and a residual class that includes any non-associated expert classes and the plurality of non-expert classes, and each EC module is configured to predict an EC predicted class based on the input data, and output the EC predicted class. The final predicted class decision module is configured to receive each MC predicted class, each MC certainty and each EC predicted class, determine a final predicted class and a final certainty based on each MC predicted class, each MC certainty and each EC predicted class, and output the final predicted class and the final certainty.

[0018] An ML model is a mathematical model that is trained by a learning process to generate an output, such as a supervisory signal, from an input, such as a feature vector. Neural networks, such as ANNs, CNNs, RNNs, BNNs, etc., as well as Support Vector Machines, Bayesian Networks, Naïve Bayes, K-Nearest Neighbor classifiers, etc., are types of ML models. For example, a supervised learning process trains an ML model using completely-labeled training data that include known input-output pairs. A semi-supervised or weakly-supervised learning process trains the ML model using incomplete training data, i.e., a small amount of labeled data (i.e., input-output pairs) and a large amount of unlabeled data (input only). An unsupervised learning process trains the ML model using unlabeled data (i.e., input only).

[0019] An ANN models the relationships between input data or signals and output data or signals using a network of interconnected nodes that is trained through a learning process. The nodes are arranged into various layers, including, for example, an input layer, one or more hidden layers, and an output layer. The input layer receives input data, such as, for example, image data, and the output layer generates output data, such as, for example, a probability that the image data contains a known object. Each hidden layer provides at least a partial transformation of the input data to the output data. A DNN has multiple hidden layers in order to model complex, nonlinear relationships between input data and output data.

[0020] In a fully-connected, feedforward ANN, each node is connected to all of the nodes in the preceding layer, as well as to all of the nodes in the subsequent layer. For example, each input layer node is connected to each hidden layer node, each hidden layer node is connected to each input layer node and each output layer node, and each output layer node is connected to each hidden layer node. Additional hidden layers are similarly interconnected. Each connection has a weight value, and each node has an activation function, such as, for example, a linear function, a step function, a sigmoid function, a tanh function, a rectified linear unit (ReLU) function, etc., that determines the output of the node based on the weighted sum of the inputs to the node. The input data propagates from the input layer nodes, through respective connection weights to the hidden layer nodes, and then through respective connection weights to the output layer nodes.

[0021] More particularly, at each input node, input data is provided to the activation function for that node, and the output of the activation function is then provided as an input data value to each hidden layer node. At each hidden layer node, the input data value received from each input layer node is multiplied by a respective connection weight, and the resulting products are summed or accumulated into an activation value that is provided to the activation function for that node. The output of the activation function is then provided as an input data value to each output layer node. At each output layer node, the output data value received from each hidden layer node is multiplied by a respective connection weight, and the resulting products are summed or accumulated into an activation value that is provided to the activation function for that node. The output of the activation function is then provided as output data. Additional hidden layers may be similarly configured to process data.

[0022] FIG. 1 depicts ANN 10, in accordance with embodiments of the present disclosure.

[0023] ANN 10 includes input layer 20, one or more hidden layers 30, 40, 50, etc., and output layer 60. Input layer 20 includes one or more input nodes 21, 22, 23, etc. Hidden layer 30 includes one or more hidden nodes 31, 32, 33, 34, 35, etc. Hidden layer 40 includes one or more hidden nodes 41, 42, 43, 44, 45, etc. Hidden layer 50 includes one or more hidden nodes 51, 52, 53, 54, 55, etc. Output layer 60 includes one or more output nodes 61, 62, etc. Generally, ANN 10 includes N hidden layers, input layer 20 includes “i” nodes, hidden layer 30 includes “j” nodes, hidden layer 40 includes “k” nodes, hidden layer 50 includes “m” nodes, and output layer 60 includes “o” nodes.

[0024] In one embodiment, N equals 3, i equals 3, j, k and m equal 5 and o equals 2 (depicted in FIG. 1). Input node 21 is coupled to hidden nodes 31 to 35, input node 22 is coupled to hidden nodes 31 to 35, and input node 23 is coupled to hidden nodes 31 to 35. Hidden node 31 is coupled to hidden nodes 41 to 45, hidden node 32 is coupled to hidden nodes 41 to 45, hidden node 33 is coupled to hidden nodes 41 to 45, hidden node 34 is coupled to hidden nodes 41 to 45, and hidden node 35 is coupled to hidden nodes 41 to 45. Hidden node 41 is coupled to hidden nodes 51

to 55, hidden node 42 is coupled to hidden nodes 51 to 55, hidden node 43 is coupled to hidden nodes 51 to 55, hidden node 44 is coupled to hidden nodes 51 to 55, and hidden node 45 is coupled to hidden nodes 51 to 55. Hidden node 51 is coupled to output nodes 61 and 62, hidden node 52 is coupled to output nodes 61 and 62, hidden node 53 is coupled to output nodes 61 and 62, hidden node 54 is coupled to output nodes 61 and 62, and hidden node 55 is coupled to output nodes 61 and 62.

[0025] Many other variations of input, hidden and output layers are clearly possible, including hidden layers that are locally-connected, rather than fully-connected, to one another.

[0026] Training an ANN includes optimizing the connection weights between nodes by minimizing the prediction error of the output data until the ANN achieves a particular level of accuracy. One method is backpropagation, or backward propagation of errors, which iteratively and recursively determines a gradient descent with respect to the connection weights, and then adjusts the connection weights to improve the performance of the network.

[0027] A multi-layer perceptron (MLP) is a fully-connected ANN that has an input layer, an output layer and one or more hidden layers. MLPs may be used for natural language processing applications, such as machine translation, speech recognition, etc. Other ANNs include recurrent neural networks (RNNs), long short-term memories (LSTMs), sequence-to-sequence models that include an encoder RNN and a decoder RNN, shallow neural networks, etc.

[0028] A CNN is a variation of an MLP that may be used for classification or recognition applications, such as image recognition, speech recognition, etc. A CNN has an input layer, an output layer and multiple hidden layers including convolutional layers, pooling layers, normalization layers, fully-connected layers, etc.

[0029] Each convolutional layer applies a sliding dot product or cross-correlation to an input volume provided by the input layer, applies an activation function to the results, and then provides the activation or output volume to the next layer. Convolutional layers typically use the ReLU function as the activation function. In some

embodiments, the activation function is provided in a separate activation layer, such as, for example, a ReLU layer.

[0030] A pooling layer reduces the dimensions of the output volume received from the preceding convolutional layer, and may calculate an average or a maximum over small clusters of data, such as, for example, 2x2 matrices. In some embodiments, a convolutional layer and a pooling layer may form a single layer of a CNN.

[0031] The fully-connected layers follow the convolutional and pooling layers, and include a flatten layer and a classification layer, followed by a normalization layer that includes a normalization function, such as the SoftMax function. The output layer follows the last fully-connected layer; in some embodiments, the output layer may include the normalization function.

[0032] Generally, classification networks, such as ANNs, CNNs, RNNs, etc., that perform pattern recognition (e.g., image, speech, activity, etc.) may be implemented in hardware, a combination of hardware and software, or software. Many classification networks predict a finite set of classes. A classification network for an autonomous vehicle may have a set of image classes that include, for example, “pedestrian,” “bicycle,” “vehicle,” “animal,” “traffic sign,” “traffic light,” “junction,” “exit,” “litter,” etc. Some of these classes are extremely important to predict in real time; otherwise, an incorrect prediction may lead to an injury or death. For example, “pedestrian,” “bicycle,” “vehicle,” etc. may be defined as important classes, while “animal,” “traffic sign,” “traffic light,” “junction,” “exit,” “litter,” etc. may not be defined as important.

[0033] FIGS. 2A and 2B depict prediction accuracy for an ANN, in accordance with embodiments of the present disclosure.

[0034] FIG. 2A depicts baseline ANN 12, according to one embodiment of the present disclosure. In one example, baseline ANN 12 received 4,999 inputs associated with 4,999 known classes, and output 4,999 predicted classes. In this example, baseline ANN 12 correctly predicted 4,975 classes and incorrectly predicted 24 classes. Since baseline ANN 12 can not distinguish between correctly predicted classes and incorrectly predicted classes, all of the predicted classes are subsequently processed the same way, which yields an accuracy of 99.7% (i.e., precision = 4,975 / 4,999). As

discussed above, it is important to have the fewest number of incorrectly predicted classes because an incorrectly predicted class can be fatal for many systems in general, and for safety-critical systems in particular.

[0035] FIG. 2B depicts certainty-based ANN 14, according to an embodiment of the present disclosure. While certainty-based ANN 14 generates a predicted class for each input and a certainty for each predicted class, baseline ANN 12 and certainty-based ANN 14 were trained using the same training methodology and data. Using the same data provided to baseline ANN 12, certainty-based ANN 14 received 4,999 inputs associated with 4,999 known classes, and output 4,999 predicted classes and 4,999 certainty values. A prediction was identified as either "certain" (i.e., "I know" this prediction is correct), or uncertain (i.e., "I don't know" whether this prediction is correct). In this embodiment, two levels of confidence are provided, i.e., a high level (positive) and a low level (negative); in other embodiments, a range of confidence levels may be provided.

[0036] Certainty-based ANN 14 correctly predicted 4,873 classes with certainty (i.e., a "true negative" condition), correctly predicted 102 classes with uncertainty (i.e., a "false positive" condition), incorrectly predicted 4 classes with certainty (i.e., a "false negative" condition), and incorrectly predicted 20 classes with uncertainty (i.e., a "true positive" condition). The false negative condition is a dangerous situation from a safety perspective. Since certainty-based ANN 14 distinguishes between certain and uncertain predicted classes, these predicted classes may be subsequently processed in different ways. In one embodiment, the uncertain predicted classes may simply be discarded, which yields an accuracy of 99.9% (e.g., precision = 4873 / (4873+4)) and a reduction in the number of incorrectly predicted classes of 83.3% (e.g., recall = 20 / (20+4)). In other embodiments, uncertain predicted classes may be re-evaluated and promoted to certain predicted classes based on predictions from additional classification networks, uncertain predicted classes may be replaced by predicted classes from additional classification networks, etc.

[0037] Importantly, because all of the certain predicted classes are subsequently processed the same way, the number of incorrectly predicted classes that are

subsequently processed has been significantly reduced from 24 classes to 4 classes, which is advantageous for many systems in general, and for safety-critical systems in particular. Determination of certainty is discussed in detail below. In some embodiments, an uncertain prediction may invoke an escalation procedure, such as, for example, ANN 14 may send a notification to a display to alert a human operator when a prediction is uncertain.

[0038] FIG. 3 depicts a block diagram of system 100, in accordance with embodiments of the present disclosure.

[0039] System 100 includes computer 102, I/O devices 142 and display 152. Computer 102 includes communication bus 110 coupled to one or more processors 120, memory 130, I/O interfaces 140, display interface 150, one or more communication interfaces 160, and one or more HAs 200. Generally, I/O interfaces 140 are coupled to I/O devices 142 using a wired or wireless connection, display interface 150 is coupled to display 152, and communication interface 160 is connected to network 162 using a wired or wireless connection. In some embodiments, certain components of computer 102 are implemented as a system-on-chip (SoC); in other embodiments, computer 102 may be hosted on a traditional printed circuit board, motherboard, etc.

[0040] In some embodiments, system 100 is an embedded system in which one or more of the components depicted in FIG. 3 are not present, such as, for example, I/O interfaces 140, I/O devices 142, display interface 150, display 152, etc. Additionally, certain components, when present, may be optimized based on various design constraints, such as, for example, power, area, etc., such as, for example, HA 200.

[0041] Communication bus 110 is a communication system that transfers data between processor 120, memory 130, I/O interfaces 140, display interface 150, communication interface 160, HAs 200, as well as other components not depicted in FIG. 3. Power connector 112 is coupled to communication bus 110 and a power supply (not shown). In some embodiments, communication bus 110 is a network-on-chip (NoC).

[0042] Processor 120 includes one or more general-purpose or application-specific microprocessors that executes instructions to perform control, computation,

input/output, etc. functions for system 100. Processor 120 may include a single integrated circuit, such as a micro-processing device, or multiple integrated circuit devices and/or circuit boards working in cooperation to accomplish the functions of processor 120. Additionally, processor 120 may include multiple processing cores, as depicted in FIG. 3. Generally, system 100 may include one or more processors 120, each containing one or more processing cores as well as various other modules.

[0043] In some embodiments, system 100 may include 2 processors 120, each containing multiple processing cores. For example, one processor 120 may be a high performance processor containing 4 “big” processing cores, e.g., Arm Cortex-A73, Cortex-A75, Cortex-A76, etc., while the other processor 120 may be a high efficiency processor containing 4 “little” processing cores, e.g., Arm Cortex-53, Arm Cortex-55, etc. In this example, the “big” processing cores include a memory management unit (MMU). In other embodiments, system 100 may be an embedded system that includes a single processor 120 with one or more processing cores, such as, for example, an Arm Cortex-M core. In these embodiments, processor 120 typically includes a memory protection unit (MPU).

[0044] In many embodiments, processor 120 may also be configured to execute classification-based machine learning (ML) models, such as, for example, ANNs, DNNs, CNNs, RNNs, SVM, Naïve Bayes, etc. In these embodiments, processor 120 may provide the same functionality as a hardware accelerator, such as HA 200. For example, system 100 may be an embedded system that does not include HA 200.

[0045] In addition, processor 120 may execute computer programs or modules, such as operating system 132, software modules 134, etc., stored within memory 130. For example, software modules 134 may include an autonomous vehicle application, a robotic application, such as, for example, a robot performing a surgical process, working with humans in a collaborative environment, etc., which may include a classification network, such as, for example, an ANN, a CNN, an RNN, a BNN, an SVM, Decision Trees, Bayesian networks, Naïve Bayes, etc.

[0046] Generally, storage element or memory 130 stores instructions for execution by processor 120 and data. Memory 130 may include a variety of non-

transitory computer-readable medium that may be accessed by processor 120. In various embodiments, memory 130 may include volatile and nonvolatile medium, non-removable medium and/or removable medium. For example, memory 130 may include any combination of random access memory (RAM), DRAM, SRAM, ROM, flash memory, cache memory, and/or any other type of non-transitory computer-readable medium.

[0047] Memory 130 contains various components for retrieving, presenting, modifying, and storing data. For example, memory 130 stores software modules that provide functionality when executed by processor 120. The software modules include operating system 132 that provides operating system functionality for system 100. Software modules 134 provide various functionality, such as image classification using CNNs, etc. Data 136 may include data associated with operating system 132, software modules 134, etc.

[0048] I/O interfaces 140 are configured to transmit and/or receive data from I/O devices 142. I/O interfaces 140 enable connectivity between processor 120 and I/O devices 142 by encoding data to be sent from processor 120 to I/O devices 142, and decoding data received from I/O devices 142 for processor 120. Generally, data may be sent over wired and/or wireless connections. For example, I/O interfaces 140 may include one or more wired communications interfaces, such as USB, Ethernet, etc., and/or one or more wireless communications interfaces, coupled to one or more antennas, such as WiFi, Bluetooth, cellular, etc.

[0049] Generally, I/O devices 142 provide input to system 100 and/or output from system 100. As discussed above, I/O devices 142 are operably connected to system 100 using a wired and/or wireless connection. I/O devices 142 may include a local processor coupled to a communication interface that is configured to communicate with system 100 using the wired and/or wireless connection. For example, I/O devices 142 may include a keyboard, mouse, touch pad, joystick, etc., sensors, actuators, etc.

[0050] Display interface 150 is configured to transmit image data from system 100 to monitor or display 152.

[0051] Communication interface 160 is configured to transmit data to and from network 162 using one or more wired and/or wireless connections. Network 162 may include one or more local area networks, wide area networks, the Internet, etc., which may execute various network protocols, such as, for example, wired and/or wireless Ethernet, Bluetooth, etc. Network 162 may also include various combinations of wired and/or wireless physical layers, such as, for example, copper wire or coaxial cable networks, fiber optic networks, Bluetooth wireless networks, WiFi wireless networks, CDMA, FDMA and TDMA cellular wireless networks, etc.

[0052] HAs 200 are configured to execute, *inter alia*, classification networks, such as, for example, ANNs, CNNs, etc., in support of various applications embodied by software modules 134. Generally, HAs 200 include one or more processors, coprocessors, processing engines (PEs), compute engines (CEs), etc., such as, for example, CPUs, GPUs, NPUs (e.g., the ARM ML Processor), DSPs, field programmable gate arrays (FPGAs), application specific integrated circuits (ASICs), controllers, microcontrollers, matrix multiplier circuits, MAC arrays, etc. HAs 200 also include a communication bus interface as well as non-volatile and/or volatile memories, such as, for example, ROM, flash memory, SRAM, DRAM, etc.

[0053] In many embodiments, HA 200 receives the ANN model and weights from memory 130 over communication bus 110 for storage in local volatile memory (e.g., SRAM, DRAM, etc.). In other embodiments, HA 200 receives a portion of the ANN model and weights from memory 130 over communication bus 110. In these embodiments, HA 200 determines the instructions needed to execute the ANN model or ANN model portion. In other embodiments, the ANN model (or ANN model portion) simply includes the instructions needed to execute the ANN model (or ANN model portion). In these embodiments, processor 120 determines the instructions needed to execute the ANN model, or, processor 120 divides the ANN model into ANN model portions, and then determines the instructions needed to execute each ANN model portion. The instructions are then provided to HA 200 as the ANN model or ANN model portion.

[0054] In further embodiments, HA 200 may store ANN models, instructions and weights in non-volatile memory. In some embodiments, the ANN model may be directly implemented in hardware using DSPs, FPGAs, ASICs, controllers, microcontrollers, adder circuits, multiply circuits, MAC circuits, etc. Generally, HA 200 receives input data from memory 130 over communication bus 110, and transmit output data to memory 130 over communication bus 110. In some embodiments, the input data may be associated with a layer (or portion of a layer) of the ANN model, and the output data from that layer (or portion of that layer) may be transmitted to memory 130 over communication bus 110.

[0055] For example, the ARM ML Processor supports a variety of ANNs, CNNs RNNs, etc., for classification, object detection, image enhancements, speech recognition and natural language understanding. The ARM ML Processor includes a control unit, a direct memory access (DMA) engine, local memory and 16 CEs. Each CE includes, *inter alia*, a MAC engine that performs convolution operations, a programmable layer engine (PLE), local SRAM, a weight decoder, a control unit, a direct memory access (DMA) engine, etc. Each MAC engine performs up to eight 16-wide dot products with accumulation. Generally, the PLE performs non-convolution operations, such as, for example, pooling operations, ReLU activations, etc. Each CE receives input feature maps (IFMs) and weights sets over the NoC and stores them in local SRAM. The MAC engine and PLE process the IFMs to generate the output feature maps (OFMs), which are also stored in local SRAM prior to transmission over the NoC.

[0056] In other embodiments, HA 200 may also include specific, dedicated hardware components that are configured to execute a pre-trained, pre-programmed, hardware-based classification network. These hardware components may include, for example, DSPs, FPGAs, ASICs, controllers, microcontrollers, multiply circuits, add circuits, MAC circuits, etc. The pre-trained, pre-programmed, hardware-based classification network receives input data, such as IFMs, and outputs one or more predictions. For hardware-based classification networks that include small ANNs, the weights, activation functions, etc., are pre-programmed into the hardware components. Generally, hardware-based classification networks provide certain benefits over more

traditional hardware accelerators that employ CPUs, GPUs, PE arrays, CE arrays, etc., such as, for example, processing speed, efficiency, reduced power consumption, reduced area, etc. However, these benefits are achieved at a price – the size of the classification network is typically small, and there is little (to no) ability to upgrade or expand the hardware components, circuits, etc. in order to update the classification network.

[0057] In many embodiments, HA 200 includes one or more processors, coprocessors, PEs, CEs, etc., that are configured to execute two or more large, main classification networks as well as one or more small, expert classification networks. In some embodiments, the expert classification networks may be pre-trained, pre-programmed, hardware-based classification networks. In these embodiments, in addition to the processors, coprocessors, PEs, CEs, etc. that are configured to execute the main classification network, HA 200 includes additional hardware components, such as DSPs, FPGAs, ASICs, controllers, microcontrollers, multiply circuits, add circuits, MAC circuits, etc., that are configured to execute each expert classification network as a separate, hardware-based classification network.

[0058] FIG. 4A depicts a block diagram of hardware accelerator 200-1, in accordance with embodiments of the present disclosure.

[0059] Generally, as discussed above, HA 200-1 may include, *inter alia*, one or more processors, coprocessors, PEs, CEs, CPUs, GPUs, NPUs, DSPs, FPGAs, ASICs, controllers, microcontrollers, matrix multiplier circuits, MAC arrays, etc., as well as a communication bus interface as well as non-volatile and/or volatile memories, such as, for example, ROM, flash memory, SRAM, DRAM, etc., a communication bus interface, etc.

[0060] HA 200-1 is configured to execute two or more main classifier (MC) modules, i.e., MC 1 module 210-1 to MC $N_M$ module 210-$N_M$, and final predicted class decision module 230. In many embodiments, $N_M$ equals 2. For clarity, the features of HA 200-1 are discussed below for embodiments including two MC modules, i.e., MC 1 module 210-1 and MC 2 module 210-2; however, these features are extendible to embodiments including three or more MC modules.

[0061] In many embodiments, MC 1 module 210-1, MC 2 module 210-2 and final predicted class decision module 230 are software modules that may be stored in local non-volatile local memory, or, alternatively, stored in memory 130 and sent to HA 200-1 via communication bus 110, as discussed above. In some embodiments, one or more of MC 1 module 210-1, MC 2 module 210-2 and final predicted class decision module 230 may be hardware-based. In other embodiments, one or more of MC 1 module 210-1, MC 2 module 210-2 and final predicted class decision module 230 may be a combination of software and hardware.

[0062] For example, MC 1 module 210-1 may include, *inter alia*, a software-based classification network, a software component that determines certainty based on an entropy calculation (discussed below), and a hardware component that performs the entropy calculation. Similarly, MC 2 module 210-2 may include, *inter alia*, a different software-based classification network, a software component that determines certainty based on an entropy calculation (discussed below), and a hardware component that performs the entropy calculation. Final predicted class decision module 230 may be a software or hardware component.

[0063] MC 1 module 210-1 includes a certainty-based classification network or main classifier 1, such as ANN 14, that receives input data sent by processor 120 via communication bus 110, and generates a predicted class and a certainty based on the input data. Similarly, MC 2 module 210-2 includes a certainty-based classification network or main classifier 2, such as ANN 14, that receives the same input data as MC 1 module 210-1, and generates a predicted class and a certainty based on the input data. The MC 1 predicted class, the MC 1 certainty, the MC 2 predicted class and the MC 2 certainty are provided to final predicted class decision module 230, which determines the final predicted class and final certainty, which are sent to processor 120 via communication bus 110. The MC 1 certainty indicates whether the MC 1 predicted class is certain or uncertain, the MC 2 certainty indicates whether the MC 2 predicted class is certain or uncertain, and the final certainty indicates whether the final predicted class is certain or uncertain.

[0064] In many embodiments, main classifier 1 and main classifier 2 are diverse classification networks, which means that main classifier 1 and main classifier 2 generate a minimal overlap of errors, e.g., incorrectly predicted classes. For example, main classifier 2 may have a slightly different ANN architecture than main classifier 1, main classifier 2 may have been trained using a different training methodology than main classifier 1, main classifier 2 may have been trained using different training data than main classifier 1, etc.; combinations of these and other factors may also be employed to create diverse classification networks.

[0065] Generally, MC 1 module 210-1 determines the MC 1 certainty based on the probability that is generated for each class. In one embodiment, the main classifier 1 is an ANN that includes an input layer, one or more hidden layers and an output layer that has a number of output nodes, and each output node generates a probability for an associated class. In many embodiments, MC 1 module 210-1 determines the MC 1 certainty by calculating the entropy of the probabilities of the associated classes; other methods for determining certainty are also contemplated. For example, the entropy may be calculated based on a sum of each output node probability times a value approximately equal to a binary logarithm of the output node probability, as given by Eq. 1.

$$entropy = -\sum_{k=0}^{n-1} p_k log_2 p_k \qquad \text{Eq. 1}$$

where $p_k$ is an output node probability determined by the Softmax function, and n is the number of nodes. Since $p_k$ has a range of values between 0 and 1, the binary logarithm of $p_k$ will be a negative number, so the sign of the sum is reversed to force the entropy to be a positive number. In many embodiments, a look up table may be used to approximate the output of the binary logarithm function, $\log_2(x)$.

[0066] MC 1 module 210-1 determines that the MC 1 certainty is certain when the entropy is less than a predetermined threshold, and uncertain when the entropy is equal to or greater than the predetermined threshold. In many embodiments, the MC 1 certainty is a binary value, the output node probabilities are between 0 and 1, and the

predetermined threshold is a fixed numeric value, such as, for example, 1. The predetermined threshold is determined during training, discussed below.

[0067] Similarly, MC 2 module 210-2 determines the MC 2 certainty based on the probability that is generated for each class. In one embodiment, the main classifier 2 is a diverse ANN that includes an input layer, one or more hidden layers and an output layer that has a number of output nodes, and each output node generates a probability for an associated class. In many embodiments, MC 2 module 210-2 determines the MC 2 certainty by calculating the entropy of the probabilities of the associated classes; other methods for determining certainty are also contemplated. MC 2 module 210-2 determines that the MC 2 certainty is certain when the entropy is less than a predetermined threshold, and uncertain when the entropy is equal to or greater than the predetermined threshold. In many embodiments, the MC 2 certainty is a binary value, the output node probabilities are between 0 and 1, and the predetermined threshold is 1. The predetermined threshold is determined during training, discussed below.

[0068] Final predicted class decision module 230 determines the final predicted class and the final certainty based on the MC 1 certainty, the MC 2 certainty, the MC 1 predicted class and the MC 2 predicted class. Advantageously, the manner in which certainty estimates from the MCs are combined to generate the final certainty is configurable both during training and inference. In many embodiments, a look-up table may be used to determine the final predicted class and the final certainty, such as, for example, Table 1; other logic mechanisms also contemplated.

| MC 1 Certainty | MC 2 Certainty | Final Certainty | Final Predicted Class | Predicted Class MC 1 = MC 2? |
|---|---|---|---|---|
| Uncertain | Uncertain | Uncertain | None | -- |
| Uncertain | Certain | Uncertain | None | -- |
| Certain | Uncertain | Uncertain | None | -- |
| Certain | Certain | Uncertain | None | No |
| | | Certain | MC 1 (MC 2) | Yes |

Table 1

[0069] More particularly, when MC 1 certainty is uncertain and MC 2 certainty is uncertain, the final certainty is uncertain and the final predicted class is indeterminate (i.e., none), which may be represented as a null value (e.g., 0), a pre-determined value indicating an indeterminate predicted class, etc. When MC 1 certainty is uncertain and MC 2 certainty is certain, the final certainty is uncertain and the final predicted class is indeterminate. When MC 1 certainty is certain and MC 2 certainty is uncertain, the final certainty is uncertain and the final predicted class is indeterminate.

[0070] When MC 1 certainty is certain and MC 2 certainty is certain, the final certainty and the final predicted class depend upon whether the MC 1 predicted class matches the MC 2 predicted class. When the MC 1 predicted class does not match the MC 2 predicted class, then the final certainty is uncertain and the final predicted class is indeterminate. When the MC 1 predicted class matches the MC 2 predicted class, then the final certainty is certain and the final predicted class is the MC 1 predicted class (which is also the MC 2 predicted class).

[0071] HA 200-1 eliminates many certain, incorrectly predicted classes (i.e., the false negative condition discussed with respect to FIG. 2B), which is advantageous from an accuracy perspective, at the expense of potentially increasing uncertain, correctly predicted classes (i.e., the false positive condition discussed with respect to FIG. 2B).

While accuracy is important, compromising functionality by reducing the number of correctly predicted classes may impact the overall efficacy of the system, degrade user experience, etc. As apparent from Table 1, for a naïve diverse system with two classification networks, all but one combination of MC 1 certainty and MC 2 certainty results in an indeterminate final predicted class, and that combination still requires that the MC 1 predicted class match the MC 2 predicted class before an actual final predicted class is output by final predicted class decision module 230.

[0072] Further embodiments of the present disclosure advantageously categorize each class predicted by the main classifiers as an expert class or a non-expert class, and include an expert classifier for each expert class. Generally, expert classes are classes that are considered to be important or critical with respect to certain aspects of the system. For example, for safety-critical systems, safety-critical classes may be selected as expert classes. The main classifiers work cooperatively with the expert classifiers to improve the resilience of the hardware accelerator, strengthen the prediction accuracy of the expert classes, detect potential hardware-related errors, etc.

[0073] FIG. 4B depicts a block diagram of hardware accelerator 200-2, in accordance with embodiments of the present disclosure.

[0074] Generally, as discussed above, HA 200-2 may include, *inter alia*, one or more processors, coprocessors, PEs, CEs, CPUs, GPUs, NPUs, DSPs, FPGAs, ASICs, controllers, microcontrollers, matrix multiplier circuits, MAC arrays, etc., as well as a communication bus interface as well as non-volatile and/or volatile memories, such as, for example, ROM, flash memory, SRAM, DRAM, etc., a communication bus interface, etc.

[0075] HA 200-2 is configured to execute two or more MC modules, i.e., MC 1 module 210-1, …, MC $N_M$ module 210-$N_M$, an expert classifier (EC) module for each expert class, i.e., EC 1 module 220-1, …, EC $N_E$ module 220-$N_E$, and final predicted class decision module 230. For clarity, the features of HA 200-2 are discussed below for embodiments including two MC modules, i.e., MC 1 module 210-1 and MC 2 module 210-2, one expert class and one EC module, i.e., EC 1 module 220-1; however,

these features are extendible to embodiments including three or more MC modules, two or more expert classes and two or more EC modules.

[0076] In many embodiments, MC 1 module 210-1, MC 2 module 210-2, EC 1 module 220-1 and final predicted class decision module 230 are software modules that may be stored in local non-volatile local memory, or, alternatively, stored in memory 130 and sent to HA 200-2 via communication bus 110, as discussed above. In some embodiments, one or more of MC 1 module 210-1, MC 2 module 210-2, EC 1 module 220-1 and final predicted class decision module 230 may be hardware-based. In other embodiments, one or more of MC 1 module 210-1, MC 2 module 210-2, EC 1 module 220-1 and final predicted class decision module 230 may be a combination of software and hardware.

[0077] For example, MC 1 module 210-1 may include, *inter alia*, a software-based classification network, a software component that determines certainty based on an entropy calculation, and a hardware component that performs the entropy calculation. Similarly, MC 2 module 210-2 may include, *inter alia*, a different software-based classification network, a software component that determines certainty based on an entropy calculation, and a hardware component that performs the entropy calculation. EC 1 module 220-1 may be a software or, preferably, a hardware component. Final predicted class decision module 230 may be a software or hardware component.

[0078] MC 1 module 210-1 includes a certainty-based classification network or main classifier 1, such as ANN 14, that receives input data sent by processor 120 via communication bus 110, and generates a predicted class and a certainty based on the input data. Similarly, MC 2 module 210-2 includes a certainty-based classification network or main classifier 2, such as ANN 14, that receives the same input data as MC 1 module 210-1, and generates a predicted class and a certainty based on the input data. In many embodiments, main classifier 1 and main classifier 2 are diverse classification networks, as discussed above.

[0079] As discussed above, MC 1 module 210-1 determines the MC 1 certainty based on the probability that is generated for each class. In one embodiment, the main

classifier 1 is an ANN that includes an input layer, one or more hidden layers and an output layer that has a number of output nodes, and each output node generates a probability for an associated class. In many embodiments, MC 1 module 210-1 determines the MC 1 certainty by calculating the entropy of the probabilities of the associated classes; other methods for determining certainty are also contemplated. For example, the entropy may be calculated based on a sum of each output node probability times a value approximately equal to a binary logarithm of the output node probability, as given by Eq. 1.

[0080] EC 1 module 220-1 includes an expert classification network that is much simpler than the main classification networks within MC 1 module 210-1 and MC 2 module 210-2. For example, EC 1 module 220-1 may include a Naïve Bayes classifier. The expert classification network may be trained using the same datasets as the main classification networks, but the labels within the training data are collapsed to the expert class label (e.g., 1) or the residual class label (e.g., 0). Because the expert classification network solves a binary classification problem, the prediction accuracy may be as high (or higher) than the main classification networks, while the cost for implementing the additional expert classification network is low, even when implemented in hardware.

[0081] The MC 1 predicted class, the MC 1 certainty, the MC 2 predicted class the MC 2 certainty and the EC 1 predicted class are provided to final predicted class decision module 230, which determines the final predicted class and final certainty, which are sent to processor 120 via communication bus 110. The MC 1 certainty indicates whether the MC 1 predicted class is certain or uncertain, the MC 2 certainty indicates whether the MC 2 predicted class is certain or uncertain, and the final certainty indicates whether the final predicted class is certain or uncertain.

[0082] Final predicted class decision module 230 determines the final predicted class and the final certainty based on the MC 1 certainty, the MC 2 certainty, the MC 1 predicted class, the MC 2 predicted class and the EC 1 predicted class. In many embodiments, a look-up table may be used to determine the final predicted class and

the final certainty, such as, for example, Table 2; other logic mechanisms are also contemplated.

| MC 1 Certainty | MC 2 Certainty | Final Certainty | Final Predicted Class | Predicted Class | |
|---|---|---|---|---|---|
| | | | | EC 1 = MC 1 or MC 2 ? | MC 1 = MC 2 ? |
| Uncertain | Uncertain | Uncertain | None | No | -- |
| | | Certain | EC 1 | Yes | -- |
| Uncertain | Certain | Uncertain | None | No | -- |
| | | Certain | EC 1 | Yes | -- |
| Certain | Uncertain | Uncertain | None | No | -- |
| | | Certain | EC 1 | Yes | -- |
| Certain | Certain | Uncertain | None | No | No |
| | | Certain | EC 1 | Yes | -- |
| | | Certain | MC1 (MC 2) | -- | Yes |

Table 2

[0083] More particularly, when MC 1 certainty is uncertain and MC 2 certainty is uncertain, the EC 1 predicted class determines the final certainty and the final predicted class. When the EC 1 predicted class is the residual class, the final certainty is uncertain and the final predicted class is indeterminate. When the EC 1 predicted class is the expert class and matches the MC 1 predicted class, the final certainty is certain and the final predicted class is the expert class. In this case, the MC 1 certainty has been promoted from uncertain to certain based on the EC 1 predicted class. When the

EC 1 predicted class is the expert class and matches the MC 2 predicted class, the final certainty is certain and the final predicted class is the expert class. In this case, the MC 2 certainty has been promoted from uncertain to certain based on the EC 1 predicted class.

[0084] When MC 1 certainty is uncertain and MC 2 certainty is certain, the EC 1 predicted class determines the final certainty and the final predicted class. When the EC 1 predicted class is the residual class, the final certainty is uncertain and the final predicted class is indeterminate. When the EC 1 predicted class is the expert class and matches the MC 1 predicted class, the final certainty is certain and the final predicted class is the expert class. In this case, the MC 1 certainty has been promoted from uncertain to certain based on the EC 1 predicted class. When the EC 1 predicted class is the expert class and matches the MC 2 predicted class, the final certainty is certain and the final predicted class is the expert class. In this case, the MC 2 certainty has been confirmed based on the EC 1 predicted class.

[0085] When MC 1 certainty is certain and MC 2 certainty is uncertain, the EC 1 predicted class determines the final certainty and the final predicted class. When the EC 1 predicted class is the residual class, the final certainty is uncertain and the final predicted class is indeterminate. When the EC 1 predicted class is the expert class and matches the MC 1 predicted class, the final certainty is certain and the final predicted class is the expert class. In this case, the MC 1 certainty has been confirmed based on the EC 1 predicted class. When the EC 1 predicted class is the expert class and matches the MC 2 predicted class, the final certainty is certain and the final predicted class is the expert class. In this case, the MC 2 certainty has been promoted from uncertain to certain based on the EC 1 predicted class.

[0086] When MC 1 certainty is certain and MC 2 certainty is certain, the final certainty and the final predicted class initially depend upon the MC 1 predicted class and the MC 2 predicted class. When the MC 1 predicted class matches the MC 2 predicted class, then the final certainty is certain and the final predicted class is the MC 1 predicted class (which is also the MC 2 predicted class). When the MC 1 predicted class does not match the MC 2 predicted class, the EC 1 predicted class then

determines the final certainty and the final predicted class. When the EC predicted class is the residual class, the final certainty is uncertain and the final predicted class is indeterminate. When the EC 1 predicted class is the expert class and matches the MC 1 predicted class, the final certainty is certain and the final predicted class is the expert class. In this case, the MC 1 certainty has been promoted from uncertain to certain based on the EC 1 predicted class. When the EC 1 predicted class is the expert class and matches the MC 2 predicted class, the final certainty is certain and the final predicted class is the expert class. In this case, the MC 2 certainty has been promoted from uncertain to certain based on the EC 1 predicted class.

[0087] In some embodiments, EC 1 module 220-1 is executed when final predicted class decision module 230 requires the EC 1 predicted class in order to determine the final certainty and the final predicted class. For the embodiments described above, EC 1 module 220-1 does not need to be executed when the MC 1 certainty and the MC 2 certainty are certain, and the MC 1 predicted class matches the MC 2 predicted class. In other embodiments, EC 1 module 220-1 is executed but only consulted when final predicted class decision module 230 requires the EC 1 predicted class in order to determine the final certainty and the final predicted class. These features are also applicable to embodiments that include multiple EC module sets.

[0088] As described above, even a single EC 1 module 220-1 advantageously reduces, to a large extent, many certain, incorrectly predicted classes. As apparent from Table 2, for an enhanced diverse system with two main classification networks and at least one expert classification network, each combination of MC 1 certainty and MC 2 certainty may result in a final predicted class. When HA 200-2 includes two or more EC modules 220-1, …, 220-$N_E$, each expert class is each assigned a unique priority (e.g., from high to low), and the logic presented in Table 2 additionally considers this priority when determining the final predicted class.

[0089] For example, in one embodiment, HA 200-2 includes two EC modules, i.e., EC 1 module 220-1 and EC module 220-2. EC 1 module 220-1 is associated with an expert class that has a high priority, while EC module 220-2 is associated with an expert class that has a low priority. The following described how the first certainty

combination of Table 2 may be modified to incorporate the additional EC module 220-2; the other certainty combinations may be similarly modified. More particularly, when the MC 1 certainty is uncertain and the MC 2 certainty is uncertain, the EC 1 and EC 2 predicted classes determine the final certainty and the final predicted class.

[0090] When the EC 1 and EC 2 predicted classes are their respective residual classes, the final certainty is uncertain and the final predicted class is indeterminate.

[0091] When the EC 1 predicted class is the high-priority expert class and the EC 1 predicted class matches the MC 1 (or MC 2) predicted class, the final certainty is certain and the final predicted class is the high-priority expert class. In this case, the MC 1 (or MC 2) certainty has been promoted from uncertain to certain based on the EC 1 predicted class; the EC 2 predicted class is not considered.

[0092] When the EC 1 predicted class is the residual class, the EC 2 predicted class is the low-priority expert class and the EC 2 predicted class matches the MC 1 (or MC 2) predicted class, the final certainty is certain and the final predicted class is the low-priority expert class. In this case, the MC 1 (or MC 2) certainty has been promoted from uncertain to certain based on the EC 2 predicted class.

[0093] Due to the low area and power requirements of each EC module 220-i, multiple, voting-redundant EC modules 220-$i_j$ and respective decision logic for each expert class may be trained and implemented in order to advantageously increase the redundancy and error tolerance.

[0094] FIG. 4C depicts a block diagram of hardware accelerator 200-3, in accordance with embodiments of the present disclosure.

[0095] Generally, as discussed above, HA 200-3 may include, *inter alia*, one or more processors, coprocessors, PEs, CEs, CPUs, GPUs, NPUs, DSPs, FPGAs, ASICs, controllers, microcontrollers, matrix multiplier circuits, MAC arrays, etc., as well as a communication bus interface as well as non-volatile and/or volatile memories, such as, for example, ROM, flash memory, SRAM, DRAM, etc., a communication bus interface, etc.

[0096] HA 200-3 is configured to execute two or more MC modules, i.e., MC 1 module 210-1, …, MC $N_M$ module 210-$N_M$, an EC module set for each expert class, i.e., EC 1 module set 224-1, …, EC $N_E$ module set 224-$N_E$, and final predicted class decision module 230. An EC module set includes two or more EC modules and an EC decision module. For example, EC 1 module set 224-1 for expert class 1 includes EC 1 module 220-$1_1$, …, EC 1 module 220-$1_i$, and EC 1 decision module 222-1. Similarly, the EC $N_E$ module set 224-$N_E$ for expert class $N_E$ includes EC $N_E$ module 220-$N_{E1}$, …, EC $N_E$ module 220-$N_{Ej}$, and EC $N_E$ decision module 222-$N_E$.

[0097] For clarity, the features of HA 200-3 are discussed below for embodiments including two MC modules, i.e., MC 1 module 210-1 and MC 2 module 210-2, one expert class and one EC module set, i.e., EC 1 module set 224-1 including EC 1 module 220-$1_1$, …, EC 1 module 220-$1_i$, and EC 1 decision module 222-1; however, these features are extendible to embodiments including three or more MC modules, two or more expert classes and two or more EC module sets.

[0098] In many embodiments, MC 1 module 210-1, MC 2 module 210-2, EC 1 module set 224-1 and final predicted class decision module 230 are software modules that may be stored in local non-volatile local memory, or, alternatively, stored in memory 130 and sent to HA 200-2 via communication bus 110, as discussed above. In some embodiments, one or more of MC 1 module 210-1, MC 2 module 210-2, EC 1 module set 224-1 and final predicted class decision module 230 may be hardware-based. In other embodiments, one or more of MC 1 module 210-1, MC 2 module 210-2, EC 1 module set 224-1 and final predicted class decision module 230 may be a combination of software and hardware.

[0099] For example, MC 1 module 210-1 may include, *inter alia*, a software-based classification network, a software component that determines certainty based on an entropy calculation, and a hardware component that performs the entropy calculation. Similarly, MC 2 module 210-2 may include, *inter alia*, a different software-based classification network, a software component that determines certainty based on an entropy calculation, and a hardware component that performs the entropy calculation. EC 1 module set 224-1 may be software or, preferably, hardware

components. Final predicted class decision module 230 may be a software or hardware component.

[0100] MC 1 module 210-1 includes a certainty-based classification network or main classifier 1, such as ANN 14, that receives input data sent by processor 120 via communication bus 110, and generates a predicted class and a certainty based on the input data. Similarly, MC 2 module 210-2 includes a certainty-based classification network or main classifier 2, such as ANN 14, that receives the same input data as MC 1 module 210-1, and generates a predicted class and a certainty based on the input data. In many embodiments, main classifier 1 and main classifier 2 are diverse classification networks, as discussed above.

[0101] As discussed above, MC 1 module 210-1 determines the MC 1 certainty based on the probability that is generated for each class. In one embodiment, the main classifier 1 is an ANN that includes an input layer, one or more hidden layers and an output layer that has a number of output nodes, and each output node generates a probability for an associated class. In many embodiments, MC 1 module 210-1 determines the MC 1 certainty by calculating the entropy of the probabilities of the associated classes; other methods for determining certainty are also contemplated. For example, the entropy may be calculated based on a sum of each output node probability times a value approximately equal to a binary logarithm of the output node probability, as given by Eq. 1.

[0102] EC 1 modules 220-$1_1$, …, 220-$1_i$ include the same the same expert classification network that is much simpler than the main classification networks within MC 1 module 210-1 and MC 2 module 210-2. For example, EC 1 modules 220-$1_1$, …, 220-$1_i$ may include a Naïve Bayes classifier. The expert classification network may be trained using the same datasets as the main classification networks, but the labels within the training data are collapsed to the expert class label (e.g., 1) or the residual class label (e.g., 0). Because the expert classification network solves a binary classification problem, the prediction accuracy may be as high (or higher) than the main classification networks, while the cost for implementing the additional expert classification network is low, even when implemented in hardware.

[0103] EC 1 decision module 222-1 determines the EC 1 predicted class based on the EC 1 predicted classes provided by EC 1 module $1_1$, ..., EC 1 module $1_i$. In some embodiments, when at least half of the EC 1 predicted classes are the expert class, EC 1 decision module 222-1 selects the expert class as the final EC predicted class. Conversely, when less than half of the EC 1 predicted classes are the expert class, EC 1 decision module 222-1 selects the residual class as the final EC 1 predicted class.

[0104] The MC 1 predicted class, the MC 1 certainty, the MC 2 predicted class the MC 2 certainty and the EC 1 predicted class are provided to final predicted class decision module 230, which determines the final predicted class and final certainty, which are sent to processor 120 via communication bus 110. The MC 1 certainty indicates whether the MC 1 predicted class is certain or uncertain, the MC 2 certainty indicates whether the MC 2 predicted class is certain or uncertain, and the final certainty indicates whether the final predicted class is certain or uncertain.

[0105] Final predicted class decision module 230 determines the final predicted class and the final certainty based on the MC 1 certainty, the MC 2 certainty, the MC 1 predicted class, the MC 2 predicted class and the EC 1 predicted class. In many embodiments, a look-up table may be used to determine the final predicted class and the final certainty, such as, for example, Table 2 (above); other logic mechanisms are also contemplated. The various combinations presented in Table 2 are discussed above.

[0106] In some embodiments, EC 1 module set 224-1 is executed when final predicted class decision module 230 requires the EC 1 predicted class in order to determine the final certainty and the final predicted class. For the embodiments described above, EC 1 module set 224-1 does not need to be executed when the MC 1 certainty and the MC 2 certainty are certain, and the MC 1 predicted class matches the MC 2 predicted class. This feature is also applicable to embodiments that include multiple EC module sets.

[0107] FIG. 5 depicts a flow diagram 300 presenting functionality for a safety-based prediction process, in accordance with an embodiment of the present disclosure.

[0108] At 310, a plurality of MC predicted classes are predicted by a plurality of MC modules based on input data. Each MC module, such as, for example, MC 1 module 210-1, MC 2 module 210-2, etc., processes a pre-trained, machine learning main classifier having at least one expert class and a plurality of non-expert classes. For example, MC 1 module 210-1 predicts the MC 1 predicted class, MC 2 module 210-2 predicts the MC 2 predicted class, etc. Additionally, each MC module determines an MC certainty. For example, MC 1 module 210-1 determines the MC 1 certainty, MC 2 module 210-2 determines the MC 2 certainty, etc., as discussed above.

[0109] At 320, an EC module associated with each expert class predicts an EC predicted class based on the input data. Each EC module, such as, for example, EC 1 module 220-1, …, EC $N_E$ module 220-$N_E$, processes a pre-trained, machine learning expert classifier that has two classes including an associated expert class and a residual class that includes any non-associated expert classes and the plurality of non-expert classes. In some embodiments discussed above, two or more EC modules are associated with each expert class, and each EC module predicts an EC predicted class.

[0110] At 330, a final predicted class and a final certainty are determined by final predicted class decision module 230 based on the MC predicted classes, the MC certainties and each EC predicted class. In some embodiments discussed above, each EC predicted class is the EC predicted class output by each EC decision module 222-1 to 222-$N_E$.

[0111] For some embodiments, the EC modules always predict EC predicted classes. In these embodiments, flow proceeds from 310 to 320, and then from 320 to 330. For other embodiments, the EC modules are only executed when final predicted class decision module 230 requires the EC predicted classes in order to determine the final certainty and the final predicted class. In these embodiments, flow proceeds from 310 to 330. If final predicted class decision module 230 requires the EC predicted classes in order to determine the final certainty and the final predicted class, then flow proceeds to 320, and then back to 330. The final predicted class is then determined by final predicted class decision module 230 based on the MC predicted classes, the MC certainties and the EC predicted classes.

[0112] At 340, the final predicted class and the final certainty is output by final predicted class decision module 230.

[0113] Generally, after the architectures of the main classifier and each expert classifier have been designed, including, for example, the input, hidden and output layers of an ANN, the convolutional, pooling, fully-connected, and normalization layers of a CNN, the fully-connected and binary activation layers of a BNN, the SVM classifiers, etc., the main classifier and each expert classifier are rendered in software in order to train the weights/parameters within the various classification layers. The resulting pre-trained main classifier and each pre-trained expert classifier may be implemented by HA 200 in several ways.

[0114] For an HA 200 that includes one or more processors, microprocessors, microcontrollers, etc., such as, for example, a GPU, a DSP, an NPU, etc., the pre-trained main classifier software implementation and each pre-trained expert classifier software implementation are adapted and optimized to run on the local processor. In these examples, the MC module, the EC modules and the final predicted class decision module are software modules. For an HA 200 that includes programmable circuitry, such as an ASIC, an FPGA, etc., the programmable circuitry is programmed to implement the pre-trained main classifier software implementation and each pre-trained expert classifier software implementation. In these examples, the MC module, the EC modules and the final predicted class decision module are hardware modules. Regardless of the specific implementation, HA 200 provides hardware-based acceleration for the main classifier and each expert classifier.

[0115] FIG. 6A depicts a block diagram of a training system 400 for a machine learning main classifier, in accordance with an embodiment of the present disclosure.

[0116] Training system 400 is a computer system that includes one or more processors, a memory, etc., that executes one or more software modules that train the main classifier included within MC 1 module 210-1, …, MC $N_M$ module 210-$N_M$. The software modules include machine learning main classifier module 410, comparison module 412 and learning module 414. In order to create a diverse classification network, each main classifier may have a different architecture, a different training

methodology, different training data, etc. For brevity, the training of the main classifier 1 within MC 1 module 210-1 is discussed below.

[0117] Initially, machine learning main classifier module 410 includes an untrained version of the main classifier included within MC 1 module 210-1. Generally, the main classifier includes one or more expert classes and several non-expert classes.

[0118] During each training cycle, machine learning main classifier module 410 receives training data (input) and determines an MC predicted class and uncertainty based on the input, comparison module 412 receives and compares the training data (expected class) to the MC predicted class and outputs error data, and learning module 414 receives the error data and the learning rate(s) for all of the classes, and determines and sends the weight adjustments to main classifier module 410. In many embodiments, the certainty is based on the entropy calculation discussed above, and the predetermined threshold is determined during training. Generally, a threshold can be determined during training by analyzing values of precision and recall in a test set and verifying whether these values conform to design specifications and acceptable safety standards.

[0119] In some embodiments, the main classifier may be trained using a single learning rate for all of the classes. A low training rate may lead to longer training times, and the main classifier might never converge successfully or provide sufficiently accurate classifications. Conversely, a high learning rate would reduce training time, but the result might be unreliable or sub-optimal. In one embodiment, learning module 414 provides a supervised learning process to train the main classifier using completely-labeled training data that include known input-output pairs. In another embodiment, learning module 414 provides a semi-supervised or weakly-supervised learning process to train the main classifier using incomplete training data, i.e., a small amount of labeled data (i.e., input-output pairs) and a large amount of unlabeled data (input only). In a further embodiment, learning module 414 provides an unsupervised learning process to train the main classifier using unlabeled data (i.e., input only).

[0120] FIG. 6B depicts a block diagram of a threshold determination process 401 for a machine learning main classifier, in accordance with an embodiment of the present disclosure.

[0121] In many embodiments, training data 405 may be divided into "train" data and "threshold" data in a particular ratio, such as, for example, 92% : 8%. While the ratio may vary, generally, the "train" data % >> "threshold" data %. The main classifier training is performed by training system 400 using the "train" data, as described above. Once training is completed, threshold determination module 416 uses the "threshold" data to determine the predetermined threshold. Inference is performed using the "threshold" data on the trained main classifier. For each sample in the "threshold" data, the entropy is calculated based on the output probabilities, which results in a range of entropy values from $entropy_{min}$ to $entropy_{max}$.

[0122] FIG. 6C depicts a block diagram of a training system 402 for a machine learning expert classifier, in accordance with an embodiment of the present disclosure.

[0123] Training system 402 is a computer system that includes one or more processors, a memory, etc., that executes one or more software modules that train each expert classifier included within each EC 1 module 220-1, …, EC $N_E$ module 220-$N_E$. The software modules include machine learning expert classifier module 420, comparison module 422 and learning module 424. Initially, machine learning expert classifier module 420 includes an untrained version of each expert classifier included within each EC 1 module 220-1, …, EC $N_E$ module 220-$N_E$. Each expert classifier includes one associated expert class and a residual class that includes any non-associated expert classes and the non-expert classes.

[0124] During each training cycle for each expert classifier, machine learning expert classifier module 420 receives training data (input) and determines an EC predicted class based on the input, comparison module 422 receives and compares the training data (expected class) to the EC predicted class and outputs error data, and learning module 424 receives the error data and the learning rate for the associated expert class, and determines and sends the weight adjustments to machine learning expert classifier module 420.

[0125] As discussed above, the main classifier may be trained using a single learning rate for all of the classes. Advantageously, the expert classifier may be trained using a learning rate that is higher than the learning rate of the main classifier due to the simplicity of the expert classifier. In some embodiments, each expert classifier associated with a particular expert class may be trained using a learning rate that is based on the priority level of the associated SC class.

[0126] In one embodiment, learning module 424 provides a supervised learning process to train the expert classifier using completely-labeled training data that include known input-output pairs. In another embodiment, learning module 424 provides a semi-supervised or weakly-supervised learning process to train the expert classifier using incomplete training data, i.e., a small amount of labeled data (i.e., input-output pairs) and a large amount of unlabeled data (input only). In a further embodiment, learning module 424 provides an unsupervised learning process to train the expert classifier using unlabeled data (i.e., input only).

[0127] Embodiments of the present disclosure advantageously provide an ensemble of classification networks that identify and reduce the number of incorrect predictions based on a level of confidence, or certainty, for each prediction. In many embodiments, a prediction may have a high level of confidence (i.e., a certain prediction) or a low level of confidence (i.e., an uncertain prediction); in other embodiments, a range of confidence levels may be provided. A certain prediction is processed normally, while an uncertain prediction is subject to additional processing that may promote the uncertain prediction to a certain prediction, may replace the uncertain prediction, discard the uncertain prediction, etc.

[0128] The embodiments described above and summarized below are combinable.

[0129] In one embodiment, a hardware accelerator includes a plurality of main classifier (MC) modules, an expert classifier (EC) module associated with each expert class, and a final predicted class decision module coupled to each MC module and each EC module. Each MC module processes a pre-trained, machine learning main classifier having at least one expert class and a plurality of non-expert classes, and

each MC module is configured to predict an MC predicted class based on input data, determine an MC certainty, and output the MC predicted class and the MC certainty. Each EC module processes a pre-trained, machine learning expert classifier having two classes including an associated expert class and a residual class that includes any non-associated expert classes and the plurality of non-expert classes, and each EC module is configured to predict an EC predicted class based on the input data, and output the EC predicted class. The final predicted class decision module is configured to receive each MC predicted class, each MC certainty and each EC predicted class, determine a final predicted class and a final certainty based on each MC predicted class, each MC certainty and each EC predicted class, and output the final predicted class and the final certainty. Each MC certainty is a binary value that indicates whether the MC predicted class is certain or uncertain, and the final certainty is a binary value that indicates whether the final predicted class is certain or uncertain.

[0130] In another embodiment of the hardware accelerator, each main classifier is an artificial neural network that includes an input layer, one or more hidden layers and an output layer having a plurality of output nodes, each output node generating a probability for an associated class; and each MC certainty is calculated based on an entropy of the probabilities of the associated classes.

[0131] In another embodiment of the hardware accelerator, the entropy is calculated based on a sum of each output node probability times a value approximately equal to a binary logarithm of the output node probability.

[0132] In another embodiment of the hardware accelerator, each MC certainty is certain when the entropy is less than a predetermined threshold, and uncertain when the entropy is equal to or greater than the predetermined threshold.

[0133] In another embodiment of the hardware accelerator, the output node probabilities are between 0 and 1, and the predetermined threshold is determined during training.

[0134] In another embodiment of the hardware accelerator, when each MC certainty indicates that the MC predicted class is certain and each MC predicted class is

the same, the final predicted class is the MC predicted class, and the final certainty indicates that the final predicted class is certain.

[0135] In another embodiment of the hardware accelerator, when each MC certainty indicates that the MC predicted class is certain, at least one MC predicted class is different, at least one MC predicted class is an expert class and at least one EC predicted class is the expert class, the final predicted class is the EC predicted class, and the final certainty indicates that the final predicted class is certain.

[0136] In another embodiment of the hardware accelerator, when at least one MC certainty indicates that the MC predicted class is uncertain, at least one MC predicted class is an expert class and at least one EC predicted class is the expert class, the final predicted class is the EC predicted class, and the final certainty indicates that the final predicted class is certain.

[0137] In another embodiment of the hardware accelerator, when each MC certainty indicates that the MC predicted class is certain and each MC predicted class is the same, each EC module does not predict and output the EC predicted class.

[0138] In one embodiment, a method includes predicting, by a plurality of main classifier (MC) modules, a plurality of MC predicted classes based on input data, each MC module processes a pre-trained, machine learning main classifier having at least one expert class and a plurality of non expert classes; determining, by each MC module, an MC certainty; predicting, by an expert classifier (EC) module associated with each expert class, an EC predicted class based on the input data, each EC module processes a pre-trained, machine learning expert classifier having two classes including an associated expert class and a residual class that includes any non-associated expert classes and the plurality of non-expert classes; determining, by a final predicted class decision module, a final predicted class and a final certainty based on each MC predicted class, each MC certainty and each EC predicted class; outputting, by the final predicted class decision module, the final predicted class and the final certainty. Each MC certainty is a binary value that indicates whether the MC predicted class is certain or uncertain, and the final certainty is a binary value that indicates whether the final predicted class is certain or uncertain.

[0139] In another embodiment of the method, each main classifier is an artificial neural network that includes an input layer, one or more hidden layers and an output layer having a plurality of output nodes, each output node generating a probability for an associated class; and said determining the MC certainty includes calculating an entropy of the probabilities of the associated classes.

[0140] In another embodiment of the method, said calculating the entropy is based on a sum of each output node probability times a value approximately equal to a binary logarithm of the output node probability.

[0141] In another embodiment of the method, each MC certainty is certain when the entropy is less than a predetermined threshold, and uncertain when the entropy is equal to or greater than the predetermined threshold.

[0142] In another embodiment of the method, the output node probabilities are between 0 and 1, and the predetermined threshold is determined during training.

[0143] In another embodiment of the method, when each MC certainty indicates that the MC predicted class is certain and each MC predicted class is the same, the final predicted class is the MC predicted class, and the final certainty indicates that the final predicted class is certain.

[0144] In another embodiment of the method, when each MC certainty indicates that the MC predicted class is certain, at least one MC predicted class is different, at least one MC predicted class is an expert class and at least one EC predicted class is the expert class, the final predicted class is the EC predicted class, and the final certainty indicates that the final predicted class is certain.

[0145] In another embodiment of the method, when at least one MC certainty indicates that the MC predicted class is uncertain, at least one MC predicted class is an expert class and at least one EC predicted class is the expert class, the final predicted class is the EC predicted class, and the final certainty indicates that the final predicted class is certain.

[0146] In another embodiment of the method, when each MC certainty indicates that the MC predicted class is certain and each MC predicted class is the same, each EC module does not predict and output the EC predicted class.

[0147] While implementations of the disclosure are susceptible to embodiment in many different forms, there is shown in the drawings and will herein be described in detail specific embodiments, with the understanding that the present disclosure is to be considered as an example of the principles of the disclosure and not intended to limit the disclosure to the specific embodiments shown and described. In the description above, like reference numerals may be used to describe the same, similar or corresponding parts in the several views of the drawings.

[0148] In this document, relational terms such as first and second, top and bottom, and the like may be used solely to distinguish one entity or action from another entity or action without necessarily requiring or implying any actual such relationship or order between such entities or actions. The terms "comprises," "comprising," "includes," "including," "has," "having," or any other variations thereof, are intended to cover a non-exclusive inclusion, such that a process, method, article, or apparatus that comprises a list of elements does not include only those elements but may include other elements not expressly listed or inherent to such process, method, article, or apparatus. An element preceded by "comprises ...a" does not, without more constraints, preclude the existence of additional identical elements in the process, method, article, or apparatus that comprises the element.

[0149] Reference throughout this document to “one embodiment,” “some embodiments,” “an embodiment,” “implementation(s),” “aspect(s),” or similar terms means that a particular feature, structure, or characteristic described in connection with the embodiment is included in at least one embodiment of the present disclosure. Thus, the appearances of such phrases or in various places throughout this specification are not necessarily all referring to the same embodiment. Furthermore, the particular features, structures, or characteristics may be combined in any suitable manner in one or more embodiments without limitation.

[0150] The term "or" as used herein is to be interpreted as an inclusive or meaning any one or any combination. Therefore, "A, B or C" means "any of the following: A; B; C; A and B; A and C; B and C; A, B and C." An exception to this definition will occur only when a combination of elements, functions, steps or acts are in some way inherently mutually exclusive. Also, grammatical conjunctions are intended to express any and all disjunctive and conjunctive combinations of conjoined clauses, sentences, words, and the like, unless otherwise stated or clear from the context. Thus, the term "or" should generally be understood to mean "and/or" and so forth. References to items in the singular should be understood to include items in the plural, and vice versa, unless explicitly stated otherwise or clear from the text.

[0151] Recitation of ranges of values herein are not intended to be limiting, referring instead individually to any and all values falling within the range, unless otherwise indicated, and each separate value within such a range is incorporated into the specification as if it were individually recited herein. The words "about," "approximately," or the like, when accompanying a numerical value, are to be construed as indicating a deviation as would be appreciated by one of ordinary skill in the art to operate satisfactorily for an intended purpose. Ranges of values and/or numeric values are provided herein as examples only, and do not constitute a limitation on the scope of the described embodiments. The use of any and all examples, or exemplary language ("e.g.," "such as," "for example," or the like) provided herein, is intended merely to better illuminate the embodiments and does not pose a limitation on the scope of the embodiments. No language in the specification should be construed as indicating any unclaimed element as essential to the practice of the embodiments.

[0152] For simplicity and clarity of illustration, reference numerals may be repeated among the figures to indicate corresponding or analogous elements. Numerous details are set forth to provide an understanding of the embodiments described herein. The embodiments may be practiced without these details. In other instances, well-known methods, procedures, and components have not been described in detail to avoid obscuring the embodiments described. The description is not to be considered as limited to the scope of the embodiments described herein.

[0153] In the following description, it is understood that terms such as “first,” “second,” “top,” “bottom,” “up,” “down,” “above,” “below,” and the like, are words of convenience and are not to be construed as limiting terms. Also, the terms apparatus, device, system, etc. may be used interchangeably in this text.

[0154] The many features and advantages of the disclosure are apparent from the detailed specification, and, thus, it is intended by the appended claims to cover all such features and advantages of the disclosure which fall within the scope of the disclosure. Further, since numerous modifications and variations will readily occur to those skilled in the art, it is not desired to limit the disclosure to the exact construction and operation illustrated and described, and, accordingly, all suitable modifications and equivalents may be resorted to that fall within the scope of the disclosure.

WHAT IS CLAIMED IS:

1. A hardware accelerator, comprising:

a plurality of main classifier (MC) modules, each MC module to process a pre-trained, machine learning main classifier having at least one expert class and a plurality of non-expert classes, each MC module configured to predict an MC predicted class based on input data, determine an MC certainty, and output the MC predicted class and the MC certainty;

an expert classifier (EC) module associated with each expert class, each EC module to process a pre-trained, machine learning expert classifier having two classes including an associated expert class and a residual class that includes any non-associated expert classes and the plurality of non-expert classes, each EC module configured to predict an EC predicted class based on the input data, and output the EC predicted class; and

a final predicted class decision module, coupled to each MC module and each EC module, configured to receive each MC predicted class, each MC certainty and each EC predicted class, determine a final predicted class and a final certainty based on each MC predicted class, each MC certainty and each EC predicted class, and output the final predicted class and the final certainty,

where each MC certainty is a binary value that indicates whether the MC predicted class is certain or uncertain, and the final certainty is a binary value that indicates whether the final predicted class is certain or uncertain.

2. The hardware accelerator according to claim 1, where:

each main classifier is an artificial neural network that includes an input layer, one or more hidden layers and an output layer having a plurality of output nodes, each output node generating a probability for an associated class; and

each MC certainty is calculated based on an entropy of the probabilities of the associated classes.

3. The hardware accelerator according to claim 2, where the entropy is calculated based on a sum of each output node probability times a value approximately equal to a binary logarithm of the output node probability.

4. The hardware accelerator according to claim 3, where each MC certainty is certain when the entropy is less than a predetermined threshold, and uncertain when the entropy is equal to or greater than the predetermined threshold.

5. The hardware accelerator according to claim 4, where the output node probabilities are between 0 and 1, and the predetermined threshold is determined during training.

6. The hardware accelerator according to claim 1, where, when each MC certainty indicates that the MC predicted class is certain and each MC predicted class is the same, the final predicted class is the MC predicted class, and the final certainty indicates that the final predicted class is certain.

7. The hardware accelerator according to claim 6, where, when each MC certainty indicates that the MC predicted class is certain, at least one MC predicted class is different, at least one MC predicted class is an expert class and at least one EC predicted class is the expert class, the final predicted class is the EC predicted class, and the final certainty indicates that the final predicted class is certain.

8. The hardware accelerator according to claim 7, where, when at least one MC certainty indicates that the MC predicted class is uncertain, at least one MC predicted class is an expert class and at least one EC predicted class is the expert class, the final predicted class is the EC predicted class, and the final certainty indicates that the final predicted class is certain.

9. The hardware accelerator according to claim 1, where, when each MC certainty indicates that the MC predicted class is certain and each MC predicted class is the same, each EC module does not predict and output the EC predicted class.

10. A method, comprising:

predicting, by a plurality of main classifier (MC) modules, a plurality of MC predicted classes based on input data, each MC module processes a pre-trained, machine learning main classifier having at least one expert class and a plurality of non-expert classes;

determining, by each MC module, an MC certainty;

predicting, by an expert classifier (EC) module associated with each expert class, an EC predicted class based on the input data, each EC module processes a pre-trained, machine learning expert classifier having two classes including an associated expert class and a residual class that includes any non-associated expert classes and the plurality of non-expert classes;

determining, by a final predicted class decision module, a final predicted class and a final certainty based on each MC predicted class, each MC certainty and each EC predicted class; and

outputting, by the final predicted class decision module, the final predicted class and the final certainty,

where each MC certainty is a binary value that indicates whether the MC predicted class is certain or uncertain, and the final certainty is a binary value that indicates whether the final predicted class is certain or uncertain.

11. The method according to claim 10, where:

each main classifier is an artificial neural network that includes an input layer, one or more hidden layers and an output layer having a plurality of output nodes, each output node generating a probability for an associated class; and

said determining the MC certainty includes calculating an entropy of the probabilities of the associated classes.

12. The method according to claim 11, where said calculating the entropy is based on a sum of each output node probability times a value approximately equal to a binary logarithm of the output node probability.

13. The method according to claim 12, where each MC certainty is certain when the entropy is less than a predetermined threshold, and uncertain when the entropy is equal to or greater than the predetermined threshold.

14. The method according to claim 13, where the output node probabilities are between 0 and 1, and the predetermined threshold is determined during training.

15. The method according to claim 10, where, when each MC certainty indicates that the MC predicted class is certain and each MC predicted class is the same, the final predicted class is the MC predicted class, and the final certainty indicates that the final predicted class is certain.

16. The method according to claim 15, where, when each MC certainty indicates that the MC predicted class is certain, at least one MC predicted class is different, at least one MC predicted class is an expert class and at least one EC predicted class is the expert class, the final predicted class is the EC predicted class, and the final certainty indicates that the final predicted class is certain.

17. The method according to claim 16, where, when at least one MC certainty indicates that the MC predicted class is uncertain, at least one MC predicted class is an expert class and at least one EC predicted class is the expert class, the final predicted class is the EC predicted class, and the final certainty indicates that the final predicted class is certain.

18. The method according to claim 10, where, when each MC certainty indicates that the MC predicted class is certain and each MC predicted class is the same, each EC module does not predict and output the EC predicted class.

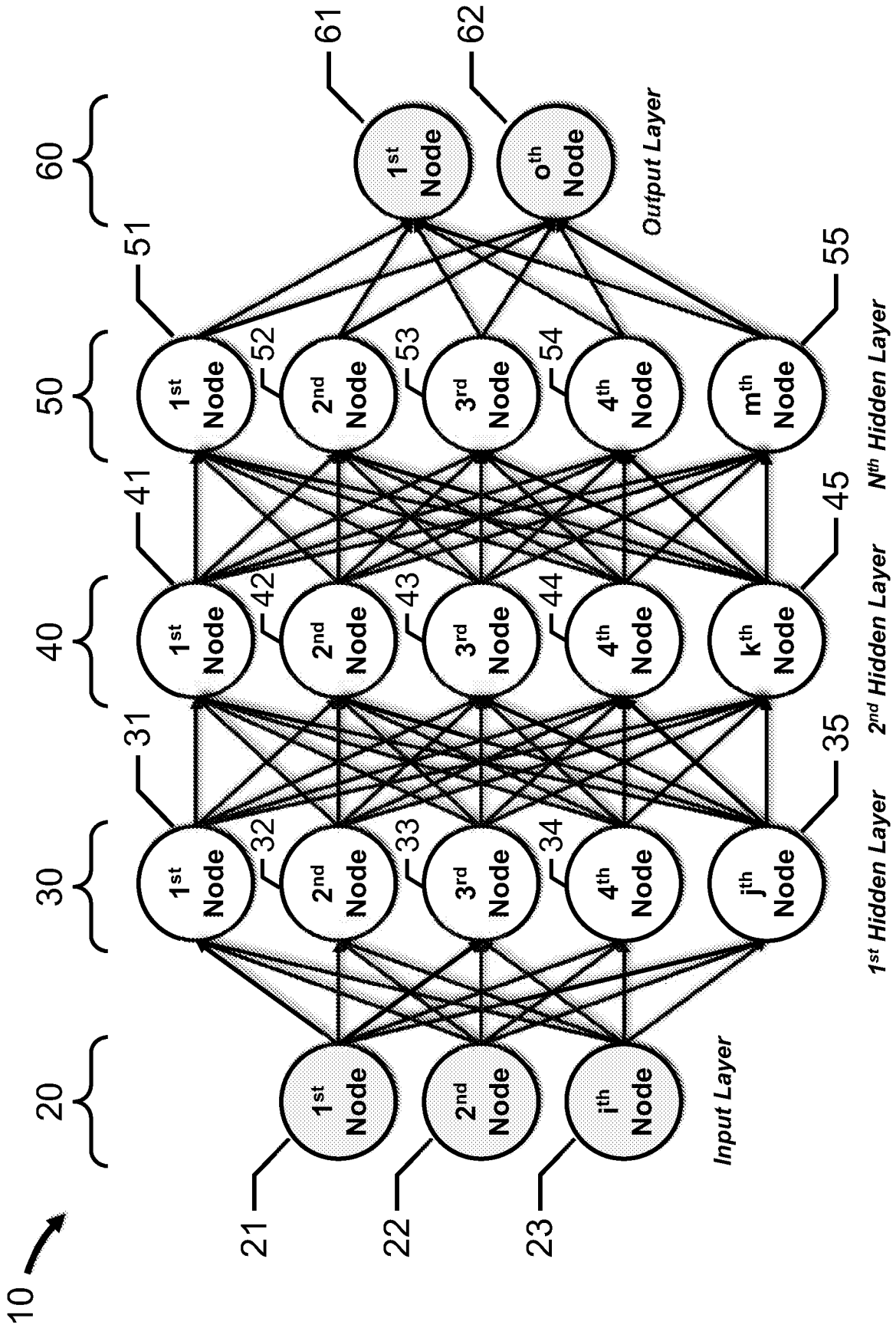
10
20
30
40
50
60
21
22
23
31
41
51
61
62
35
45
55
1st Node
2nd Node
3rd Node
4th Node
ith Node
jth Node
kth Node
mth Node
oth Node
Input Layer
1st Hidden Layer
2nd Hidden Layer
Nth Hidden Layer
Output Layer

FIG. 1

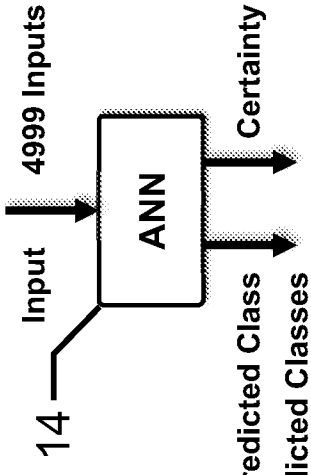
Input
4999 Inputs
ANN
Predicted Class
4999 Predicted Classes
12

FIG. 2A


Input
4999 Inputs
ANN
Certainty
Predicted Class
4999 Predicted Classes
14
Certainty
Uncertain
Certain
Incorrectly Predicted Class
Correctly Predicted Class
True Positive 20
False Positive 102
False Negative 4
True Negative 4873

FIG. 2B

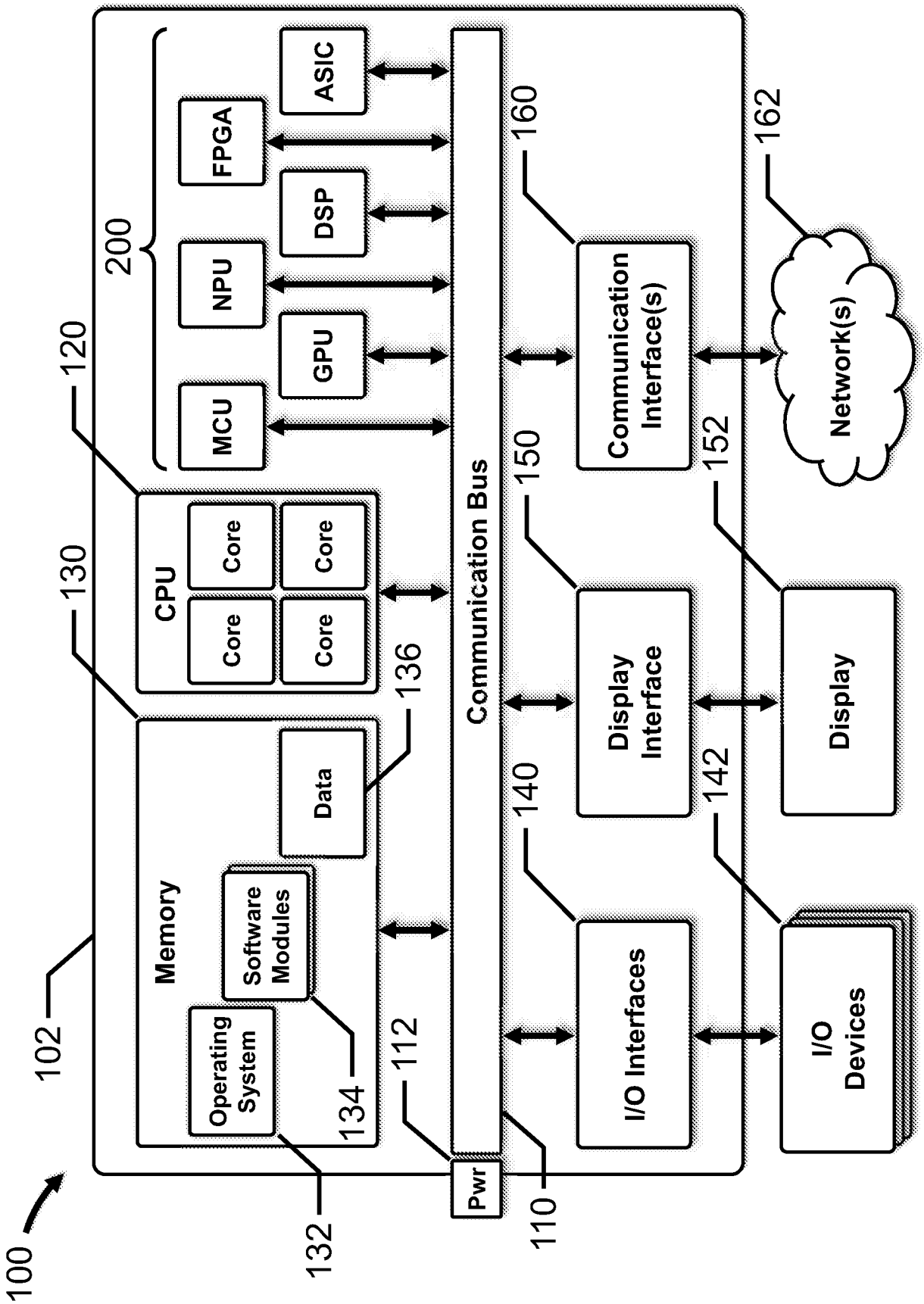
100
102
120
130
132
110
112
134
136
140
142
150
152
160
162
200
Memory
Operating System
Software Modules
Data
CPU
Core
Core
Core
Core
MCU
NPU
GPU
DSP
FPGA
ASIC
Pwr
Communication Bus
I/O Interfaces
Display Interface
Communication Interface(s)
I/O Devices
Display
Network(s)

FIG. 3

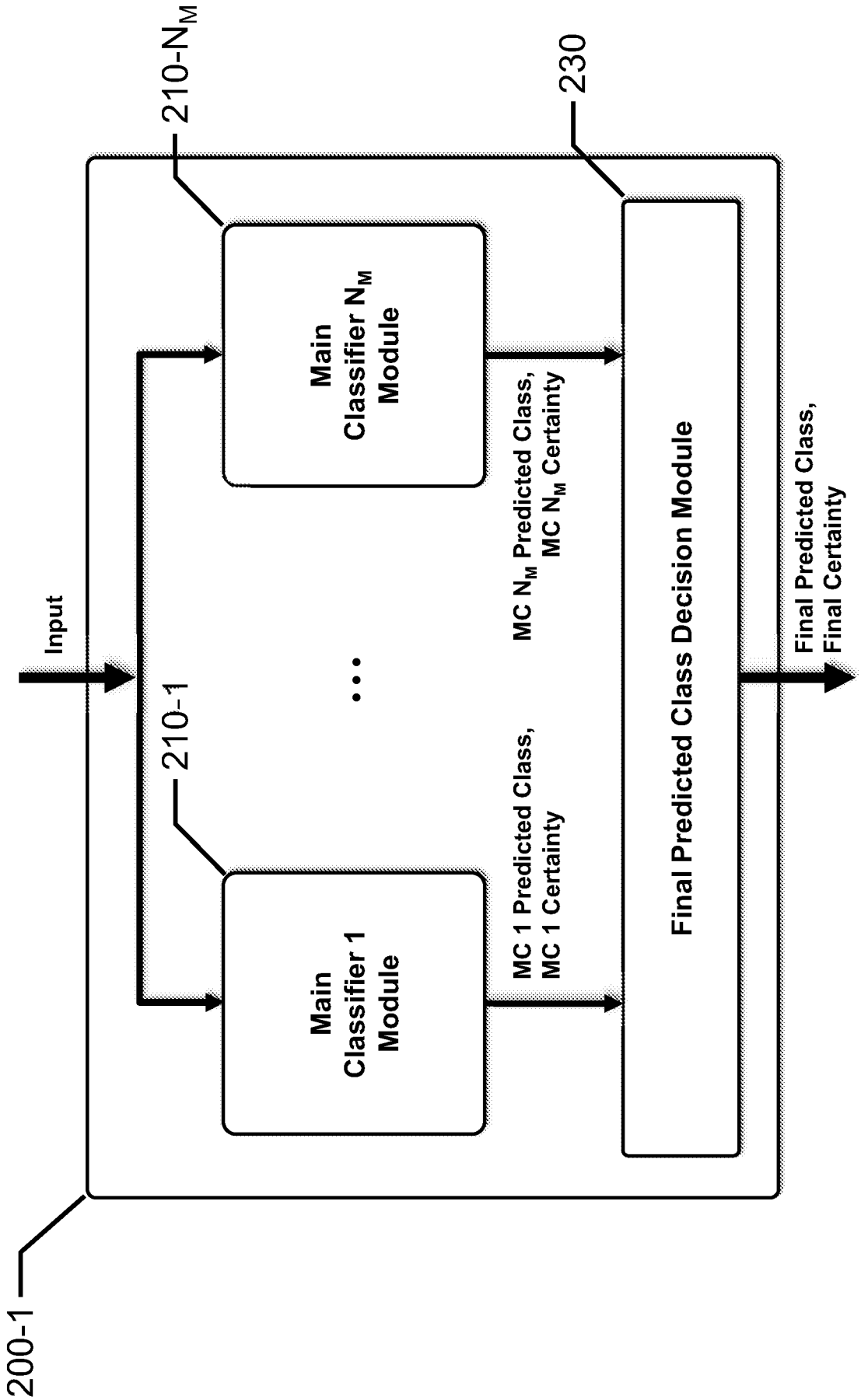
200-1
Input
210-1
210-N_M
230
Main Classifier 1 Module
Main Classifier N_M Module
MC 1 Predicted Class, MC 1 Certainty
MC N_M Predicted Class, MC N_M Certainty
Final Predicted Class Decision Module
Final Predicted Class, Final Certainty

FIG. 4A

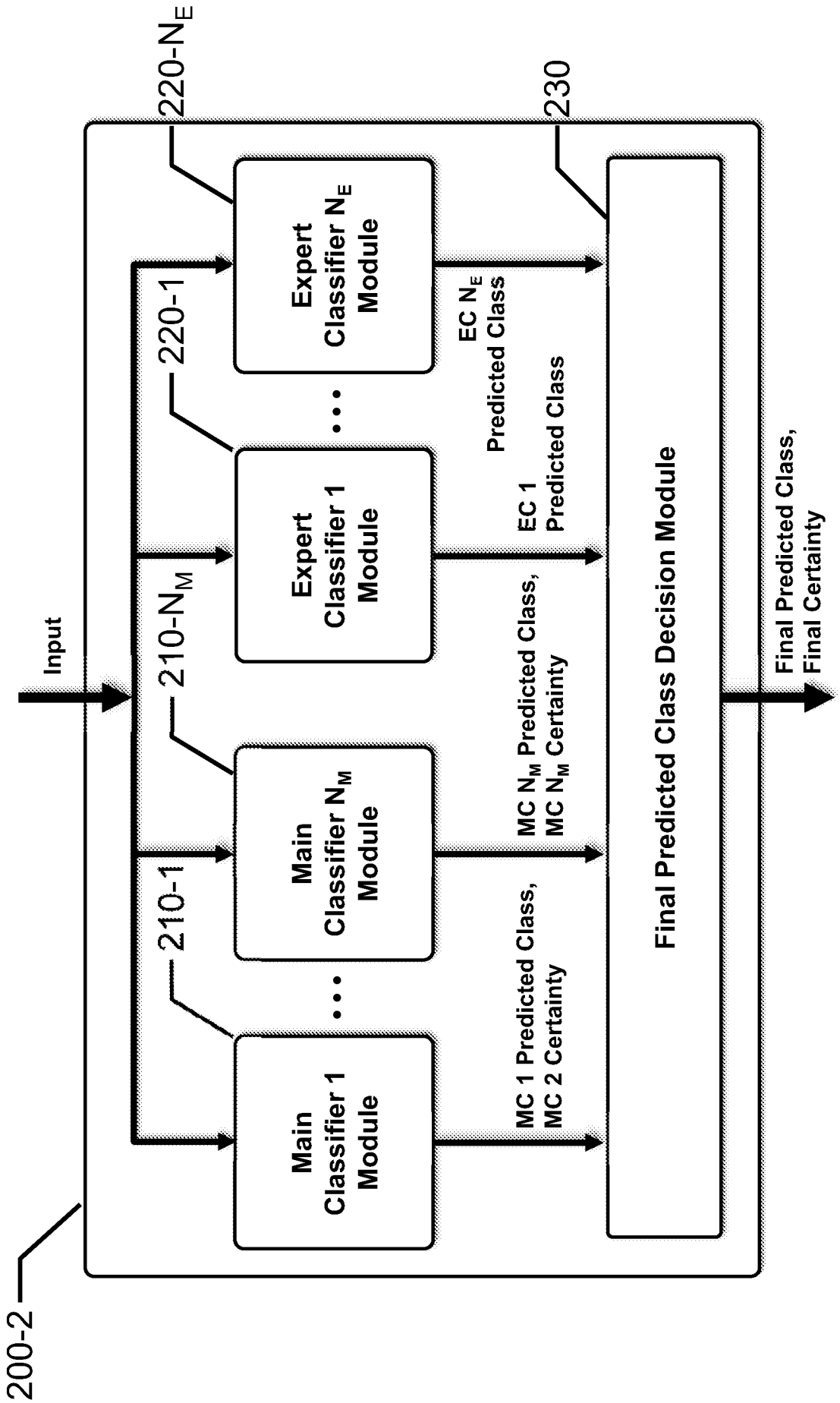
200-2
Input
210-1
210-N_M
220-1
220-N_E
230
Main Classifier 1 Module
Main Classifier N_M Module
Expert Classifier 1 Module
Expert Classifier N_E Module
MC 1 Predicted Class, MC 2 Certainty
MC N_M Predicted Class, MC N_M Certainty
EC 1 Predicted Class
EC N_E Predicted Class
Final Predicted Class Decision Module
Final Predicted Class, Final Certainty

FIG. 4B

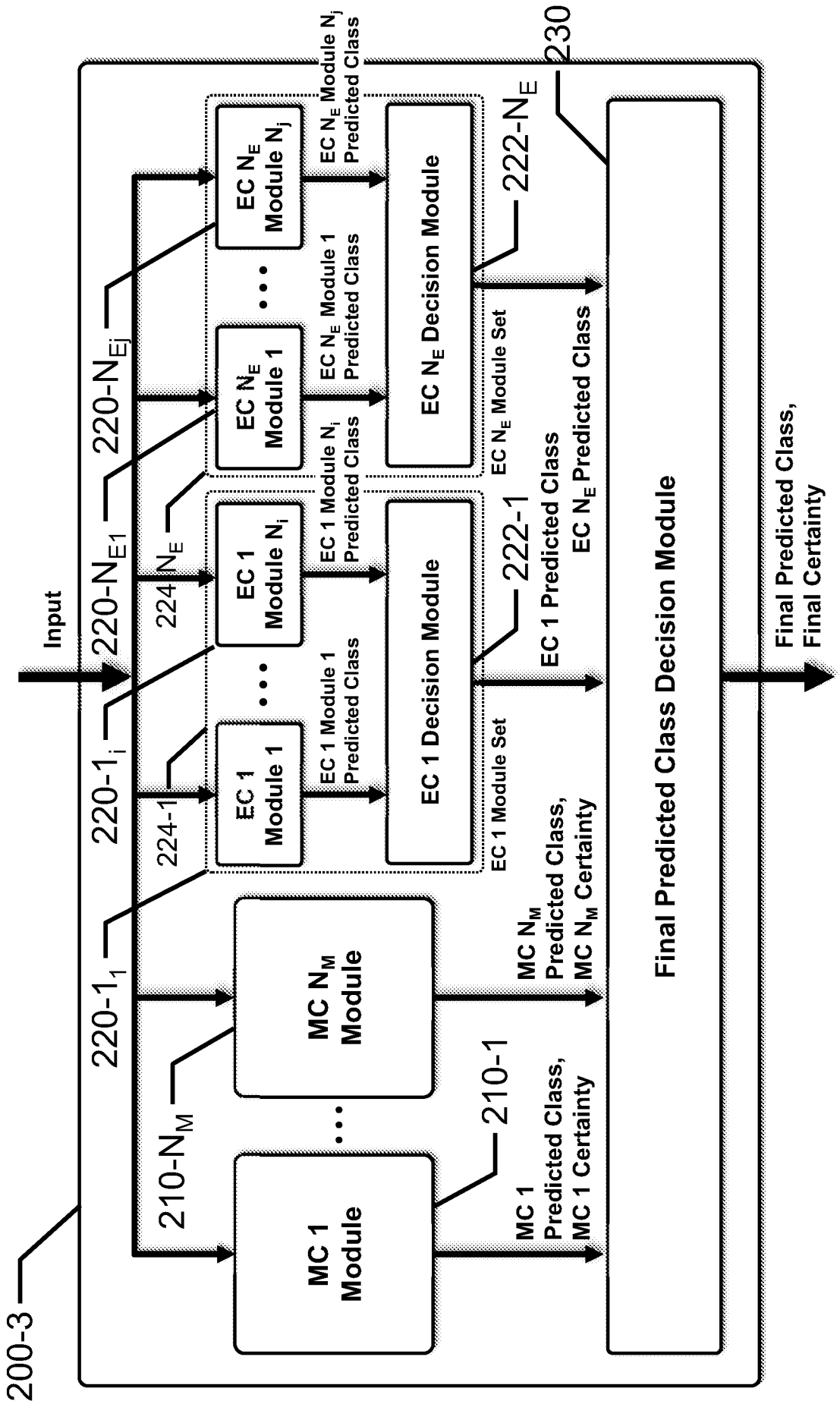
200-3
Input
220-1$_1$
220-1$_i$
220-N$_{E1}$
220-N$_{Ej}$
224-1
224-N$_E$
210-1
210-N$_M$
MC 1 Module
MC N$_M$ Module
EC 1 Module 1
EC 1 Module N$_i$
EC N$_E$ Module 1
EC N$_E$ Module N$_j$
EC 1 Module 1 Predicted Class
EC 1 Module N$_i$ Predicted Class
EC N$_E$ Module 1 Predicted Class
EC N$_E$ Module N$_j$ Predicted Class
EC 1 Decision Module
EC N$_E$ Decision Module
EC 1 Module Set
EC N$_E$ Module Set
222-1
222-N$_E$
EC 1 Predicted Class
EC N$_E$ Predicted Class
MC 1 Predicted Class, MC 1 Certainty
MC N$_M$ Predicted Class, MC N$_M$ Certainty
230
Final Predicted Class Decision Module
Final Predicted Class, Final Certainty

FIG. 4C

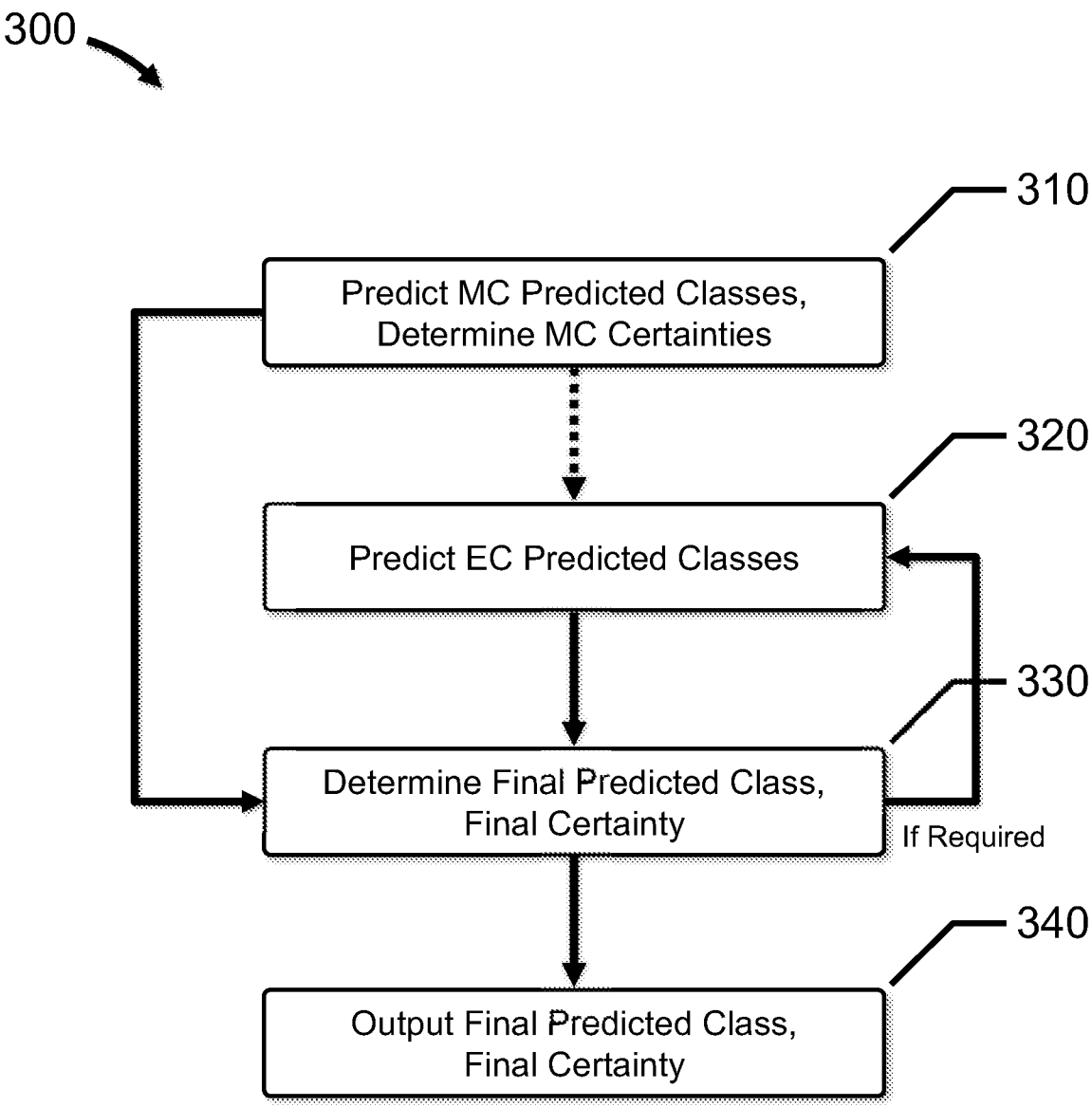
300
310
Predict MC Predicted Classes,
Determine MC Certainties
320
Predict EC Predicted Classes
330
Determine Final Predicted Class,
Final Certainty
If Required
340
Output Final Predicted Class,
Final Certainty

FIG. 5

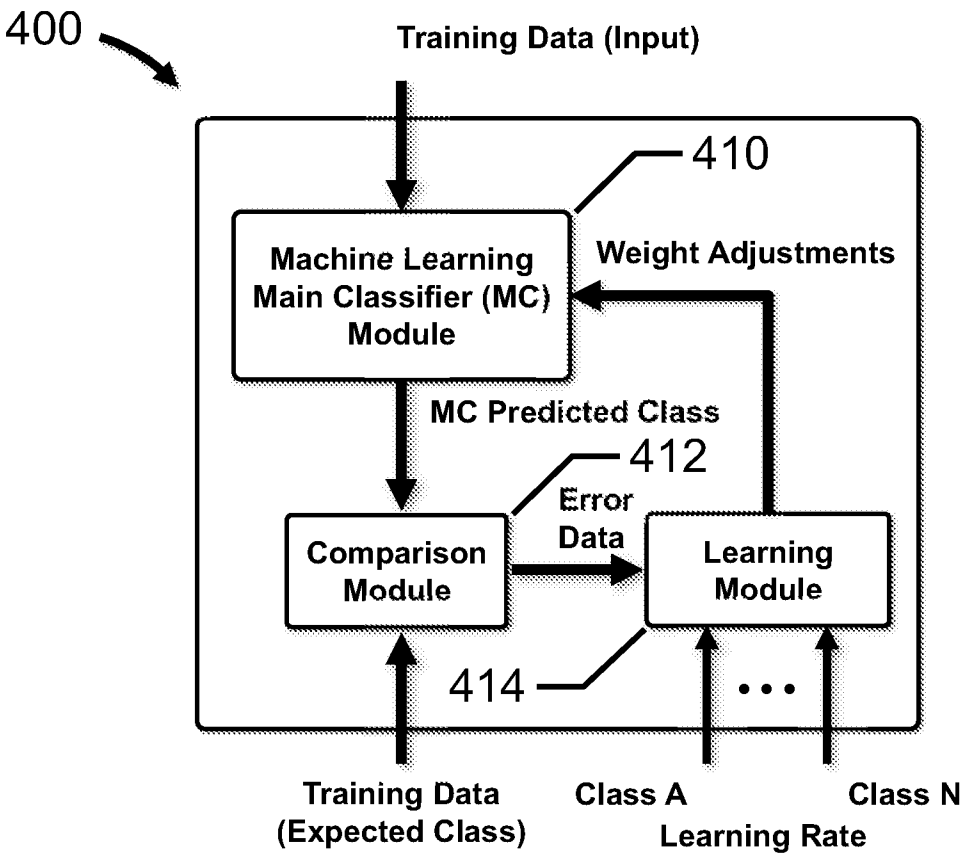
400
Training Data (Input)
410
Machine Learning Main Classifier (MC) Module
Weight Adjustments
MC Predicted Class
412
Error Data
Comparison Module
Learning Module
414
Training Data (Expected Class)
Class A
Class N
Learning Rate

FIG. 6A

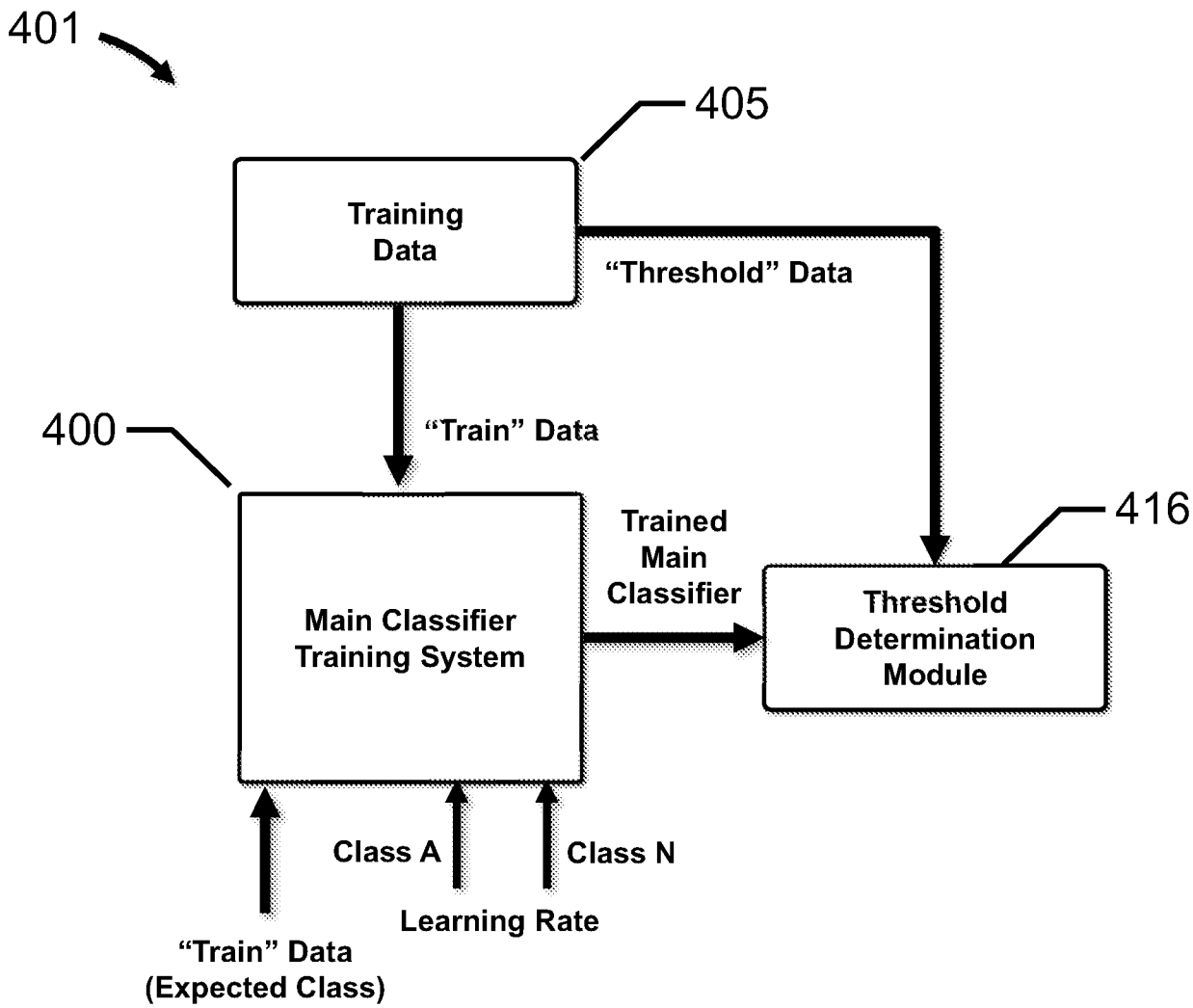
401
405
Training Data
"Threshold" Data
"Train" Data
400
Trained Main Classifier
416
Main Classifier Training System
Threshold Determination Module
Class A
Class N
Learning Rate
"Train" Data (Expected Class)

FIG. 6B

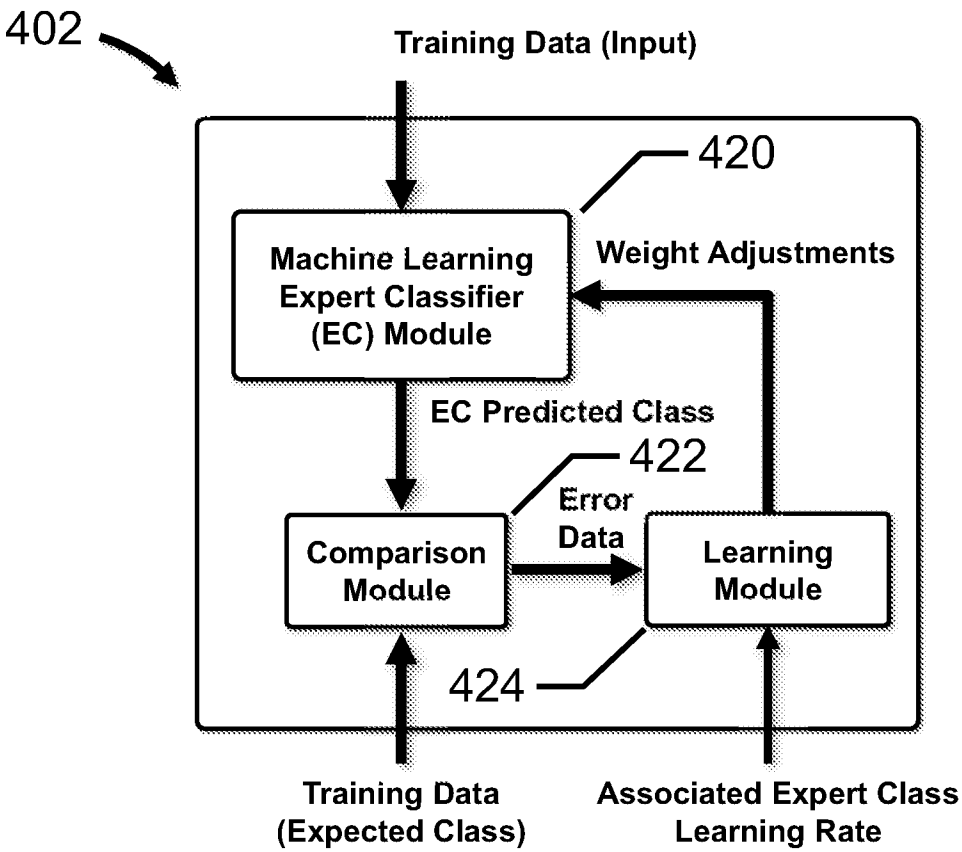
402
Training Data (Input)
420
Machine Learning Expert Classifier (EC) Module
Weight Adjustments
EC Predicted Class
422
Error Data
Comparison Module
Learning Module
424
Training Data (Expected Class)
Associated Expert Class Learning Rate

FIG. 6C