



(51) International Patent Classification:

G06K 9/00 (2006.01) *G06T 17/00* (2006.01)
G06K 9/62 (2006.01) *G10L 15/26* (2006.01)

(21) International Application Number:

PCT/US2016/020869

(22) International Filing Date:

4 March 2016 (04.03.2016)

(25) Filing Language:

English

(26) Publication Language:

English

(30) Priority Data:

62/128,393 4 March 2015 (04.03.2015) US
62/222,676 23 September 2015 (23.09.2015) US

(71) Applicant: **THE REGENTS OF THE UNIVERSITY OF CALIFORNIA** [US/US]; 1111 Franklin Street, Twelfth Floor, Oakland, CA 94607 (US).

(72) Inventors: **TU, Zhuowen**; 1111 Franklin Street, 5th Floor, Oakland, CA 94607-5200 (US). **LEE, Chen-yu**; 1111 Franklin Street, 5th Floor, Oakland, CA 94607-5200 (US).

(74) Agents: **ZHANG, Li (lily)** et al.; Mintz Levin Cohn Ferris Glovsky And Popeo, P.C., 3580 Carmel Mountain Road, Suite 300, San Diego, CA 92130-6768 (US).

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

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

Published:

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

(54) Title: CONVOLUTIONAL NEURAL NETWORK WITH TREE POOLING AND TREE FEATURE MAP SELECTION

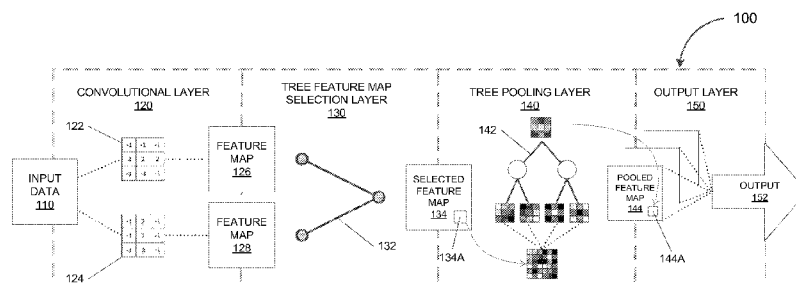


FIG. 1

(57) Abstract: In one aspect, there is provided a method for training a convolutional neural network. The method may include: receiving training data; utilizing the training data to train a convolutional neural network comprising a tree pooling layer, wherein the tree pooling layer applies a soft decision tree to generate one or more pooled feature map; providing a trained convolutional neural network comprising a tree pooling layer. Related systems, methods, and articles of manufacture are also disclosed.

CONVOLUTIONAL NEURAL NETWORK WITH TREE POOLING AND TREE FEATURE MAP SELECTION

CROSS REFERENCE TO RELATED APPLICATIONS

[0001] This application claims priority to U.S. Provisional Patent Application No. 62/128,393 filed March 4, 2015, entitled “FOREST CONVOLUTIONAL NEURAL NETWORKS” and U.S. Provisional Patent Application No. 62/222,676, filed September 23, 2015, entitled “GENERALIZING POOLING FUNCTIONS IN CONVOLUTIONAL NEURAL NETWORKS,” the contents of both applications are hereby incorporated by reference in their entirety.

STATEMENT OF GOVERNMENT SPONSORED SUPPORT

[0002] Certain aspects of the present disclosure were developed with U.S. Government Support under Grant No. NSF IIS-1360566 and NSF IIS-1360568 awarded by the National Science Foundation. The U.S. Government has certain rights in the subject matter of the present disclosure.

TECHNICAL FIELD

[0003] The subject matter disclosed herein relates to machine learning and more specifically to neural networks.

BACKGROUND

[0004] One of the foremost objectives in the development of artificial intelligence (AI) is to create a machine analog of the human brain. Ideally, AI should exhibit the ability to process

complex data and evolve through learning. A convolutional neural network is one type of machine learning architecture that endeavors to emulate human perception and cognition. For example, a convolutional neural network can be used to perform tasks such as facial recognition, image search, speech recognition and translation, disease classification, and bio-marker discovery. A convolutional neural network can include multiple convolutional layers. At each convolutional layer, individual convolutional kernels are applied to data (e.g., image, speech, genome) to yield a number of feature maps. A convolutional kernel operates, in some respects, operates like a filter that detects a specific feature (e.g., lines, shapes, objects) in the data. As such, each feature map associated with the data may depict one or more occurrences of one particular feature in the data. The convolutional neural network can also include multiple pooling layers that alternate with the convolutional layers. At a pooling layer, each portion of a feature map from a preceding convolutional layer is subject to a pooling function. For example, a maximum function or an average function is typically applied to every portion of the feature map. This generates a pooled feature map that is a subsample of that feature map.

SUMMARY

[0005] Methods, systems, and apparatus, including computer program products, are provided for training a convolutional neural network having a tree pooling layer that applies a soft decision tree to generate pooled feature maps. In some example embodiments, there is provided a method that includes receiving training data; utilizing the training data to train a convolutional neural network comprising a tree pooling layer, wherein the tree pooling layer applies a soft decision tree to generate one or more pooled feature map; and providing a trained convolutional neural network comprising a tree pooling layer

[0006] In some variations, one or more of the features disclosed herein including the following features can optionally be included in any feasible combination. The convolutional neural network may include a convolutional layer configured to generate a plurality of feature maps based on the training data. The convolutional layer may generate a feature map by applying a convolutional kernel to the training data. The convolutional kernel may be adapted to detect a feature in the training data. The feature map may depict one or more occurrences of the feature in the training data.

[0007] The convolutional neural network may further include a tree feature map selection layer configured to generate at least one selected feature map based on the plurality of feature maps generated at the convolutional layer. The tree feature map selection layer may be configured to apply a soft decision tree to generate the at least one selected feature map. The soft decision tree may combine two or more of the plurality of the feature maps into the at least one selected feature map. The soft decision tree combines the two or more feature maps according to a mixing proportion which may indicate a portion of each of the two or more feature maps to include in the selected feature map.

[0008] The tree pooling layer may be configured to apply the soft decision tree to each portion of the selected feature map to generate a corresponding portion of the pooled feature map. The soft decision tree may include a plurality of leaf nodes and decision nodes, wherein each leaf node corresponds to a pooling filter to apply to a portion of the selected feature map, and wherein a decision node applies a soft splitting function that combines an output from each child node of that decision node according to a mixing proportion. The pooling filter may include one of a maximum operation, an average operation, and a stochastic operation. The

mixing proportion may indicate a portion of the output from each child node to include in a combination of the outputs from the child nodes.

[0009] The convolutional neural network may further include an output layer configured to generate a training output based on the one or more pooled feature maps. Training the convolutional neural network may include determining, by backpropagation and gradient descent, one or more optimizations based on an error associated with the training output. Providing the trained convolutional neural network may include sending and/or storing the trained convolutional neural network.

[00010] Methods, systems, and apparatus, including computer program products, are provided for utilizing a trained convolutional neural network having a tree pooling layer that applies a soft decision tree to generate pooled feature maps. In some example embodiments, there is provided a method that includes receiving input data; processing the input data by utilizing a trained convolutional neural network comprising a tree pooling layer, wherein the tree pooling layer applies a soft decision tree to generate one or more pooled feature maps; and providing, as an output, a result of the processing performed by the trained convolutional neural network.

[00011] In some variations, one or more of the features disclosed herein including the following features can optionally be included in any feasible combination. The trained convolutional neural network may further include a convolutional layer, wherein the convolutional layer is configured to generate a plurality of feature maps based on the input data, and wherein the convolutional layer generates each of the plurality of feature maps by applying a convolutional kernel to the input data. The trained convolutional neural network may further include a tree feature map selection layer, wherein the tree feature map selection layer is

configured to apply a soft decision tree to generate at least one selected feature map, and wherein the soft decision tree generates the at least one feature map by combining two or more of the plurality of feature maps generated at the convolutional layer.

DESCRIPTION OF THE DRAWINGS

[00012] The accompanying drawings, which are incorporated in and constitute a part of this specification, show certain aspects of the subject matter disclosed herein and, together with the description, help explain some of the principles associated with the disclosed implementations. In the drawings,

[00013] FIG. 1 depicts an example of a convolutional neural network, in accordance with some example embodiments;

[00014] FIG. 2 depicts a system diagram illustrating a system, in accordance with some example embodiments;

[00015] FIG. 3 depicts a flowchart illustrating a process for training a convolutional neural network, in accordance with some example embodiments;

[00016] FIG. 4 depicts a flowchart illustrating a process for training a convolutional neural network, in accordance with some example embodiments;

[00017] FIG. 5 depicts an example of a soft decision tree, in accordance with some embodiments; and

[00018] FIG. 6 depicts a flowchart illustrating a process for utilizing a trained convolutional neural network, in accordance with some example embodiments.

DETAILED DESCRIPTION

[00019] In some example embodiments, a convolutional neural network may be configured to include a tree feature map selection layer where feature maps from a preceding convolutional layer may be combined to generate selected feature maps. At the tree feature map selection layer, soft decision trees may be applied to make “soft” selections across multiple feature maps. A “soft” selection across two or more feature maps may combine the feature maps into a single selected feature map. As such, a selected feature map may represent a combination of two or more feature maps from the preceding convolutional layer.

[00020] In some example embodiments, a convolutional neural network may be configured to include a tree pooling layer at which multiple pooling filters may be applied to a feature map (e.g., from a preceding convolutional layer) or a selected feature map (e.g., from a preceding tree feature map selection layer). At the tree pooling layer, soft decision trees may be applied to make “soft” decisions over outputs from the different pooling filters. Each of the pooling filters may apply a different pooling operation to subsample a portion of the feature map. A “soft” decision may combine the outputs from two or more pooling filters to generate a corresponding portion of a pooled feature map. As such, the pooled feature map may represent a subsampling of the corresponding feature map.

[00021] In some example embodiments, a convolutional neural network having a tree feature map selection layer and/or a tree pooling layer may be trained. For example, a convolutional neural network may be trained in a supervised learning mode including backpropagation and gradient descent.

[00022] In some example embodiments, a trained convolutional neural network having a tree feature map selection layer and/or a tree pooling layer may be used to process input data. For example, a trained convolutional neural network may be used to process image, speech,

genomic data, and/or any other type of data. The output of a trained convolutional neural network may be, for example, a classification of the input data.

[00023] FIG. 1 depicts an example of a convolutional neural network 100, in accordance with some example embodiments. Referring to FIG. 1, the convolutional neural network 100 may include a plurality of layers including a convolutional layer 120, a tree feature map selection layer 130, a tree pooling layer 140, and an output layer 150.

[00024] In some example embodiments, the convolutional neural network 100 receives input data 110 at the convolutional layer 120. At the convolutional layer 120, a plurality of convolutional kernels may be applied to the input data 110 including a first convolutional kernel 122 and a second convolutional kernel 124. For example, the first convolutional kernel 122 may be applied to the input data 110 to generate a first feature map 126 and the second convolutional kernel 124 may be applied to the input data 110 to generate a second feature map 128. A different number of feature maps may be generated at the convolutional layer 120 without departing from the scope of the present disclosure. For instance, additional convolutional kernels may be applied to the input data 110 at the convolutional layer 120 to generate additional feature maps.

[00025] Each convolutional kernel may process data like a filter that is adapted to detect a specific feature in the input data 110. For example, where the input data 110 represents an image, the first convolutional kernel 122 may be adapted to detect horizontal lines in the input data 110 while the second convolutional kernel 122 may be adapted to detect vertical lines in the input data 110. As such, the first feature map 126 may depict all instances of horizontal lines in the image and the second feature map 128 may depict all instances of vertical lines in the image. The first convolutional kernel 122 and the second convolutional kernel 122 may also be adapted

to detect more complex features in the input data 110 including shapes and objects (e.g., facial features).

[00026] At the tree feature map selection layer 130, a first soft decision tree 132 may be applied to the feature maps generated at the convolutional layer 120 including the first feature map 126 and the second feature map 128. In some example embodiments, the soft decision tree 132 makes a “soft” selection across the first feature map 126 and the second feature map 128. The “soft” selection may combine the first feature map 126 and the second feature map 128 into a selected feature map 134. The first feature map 126 and the second feature map 128 may be combined according to a mixing proportion. The mixing proportion may indicate a portion (e.g., percentage) of each of the first feature map 126 and the second feature map 128 to include in the selected feature map 134. In some example embodiments, the mixing proportion may vary based on the first feature map 126 and the second feature map 128.

[00027] A second soft decision tree 142 may be applied to the selected feature map 134 at the tree pooling layer 140. The second soft decision tree 142 may be applied to individual portions (e.g., a first portion 134A) of the selected feature map 134 to generate corresponding portions (e.g., a second portion 144A) in a pooled feature map 144. Different pooling filters (e.g., maximum pooling filter, average pooling filter, stochastic pooling filter, and/or the like) may be applied to the first portion 134A. The second soft decision tree 142 may generate the second portion 144A by combining the outputs from the different pooling filters according to a mixing proportion that may vary based on the first portion 134A of the selected feature map 134. In some example embodiments, the pooled feature map 144 may be a sub-sample of the selected feature map 134 that maintains salient features from the selected feature map 134 while

mitigating any noise. As such, the selected feature map 134 may provide a more robust and compact representation of the selected feature map 134.

[00028] The output layer 150 may provide an output 152 that is generated based on a plurality of pooled feature maps including the pooled feature map 144. In some example embodiments, the convolutional neural network 100 may be trained based on the output 152. For example, training the convolutional neural network 100 may include generating a loss function representative of an error associated with the output 152 as back propagated from the output layer 150 through to the convolutional layer 120. Training the convolutional neural network 100 may further include determining one or more optimizations by performing gradient descent to minimize the loss function.

[00029] The convolutional neural network 100 may include additional and/or different layers without departing from the scope of the present disclosure. For example, in some example embodiments, the convolutional neural network 100 may include additional tree feature map selection layers and/or tree pooling layers. The convolutional neural network 100 may optionally include one or more conventional pooling layers in addition to one or more tree pooling layers (e.g., the tree pooling layer 140) without departing from the scope of the present disclosure. In some embodiments where the convolutional neural network 100 includes both conventional and tree pooling layers, tree pooling layers (e.g., the tree pooling layer 140) may occupy lower levels of the convolutional neural network 100 relative to conventional tree pooling layers which may occupy higher levels of the convolutional neural network 100.

[00030] FIG. 2 depicts a system diagram illustrating a system 200, in accordance with some example embodiments. Referring to FIGS. 1-2, the convolutional neural network 100 may be implemented using the system 200. In some example embodiments, the system 200 may be

realized in digital electronic circuitry, integrated circuitry, specially designed application specific integrated circuits (ASICs), field programmable gate arrays (FPGAs) computer hardware, firmware, software, and/or combinations thereof.

[00031] In some example embodiments, the convolutional neural network system 200 may include one or more processors that implement a plurality of modules including a convolutional module 210, a tree feature map selection module 212, a tree pooling module 214, and an output module 216. The convolutional neural network 200 may include additional and/or different modules without departing from the scope of the present disclosure.

[00032] The convolutional module 210 may be configured to implement one or more convolutional layers (e.g., the convolutional layer 120) of the convolutional neural network 100. For example, the convolutional module 210 may receive the input data 110 and apply a plurality of convolutional kernels (e.g., the first convolutional kernel 122 and the second convolutional kernel 124) to generate a plurality of feature maps (e.g., the first feature map 126 and the second feature map 128).

[00033] The tree feature map selection module 212 may be configured to implement one or more tree feature map selection layers (e.g., the tree feature map selection layer 130) of the convolutional neural network 100. For example, the tree feature map selection module 212 may apply one or more soft decision trees (e.g., the first soft decision tree 132) to a plurality of feature maps (e.g., the first feature map 126 and the second feature map 128) to generate one or more selected feature maps (e.g., the selected feature map 134).

[00034] The tree pooling module 214 may be configured to implement one or more tree pooling layers (e.g., the tree pooling layer 140) of the convolutional neural network 100. For example, the tree pooling module 214 may apply one or more soft decision trees (e.g., the second

soft decision tree 142) to portions (e.g., the first portion 134A) of each selected feature map (e.g., the selected feature map 134) to generate a corresponding pooled feature map (e.g., the pooled feature map 144).

[00035] The output module 216 may be configured to implement the output layer (e.g., the output layer 150) of the convolutional neural network 100. For example, the output module 216 may provide an output (e.g., the output 152) based on one or more pooled feature maps (e.g., the pooled feature map 144).

[00036] In some example embodiments, the system 200 may be configured to communicate with a device 220 (e.g., a personal computer, workstation, smartphone) via a wired and/or wireless network 230. The device 220 may provide a user interface for interacting with the system 200 including to train the system 200 and/or to utilize the system 200 to process input data. For instance, a user may provide, via the device 220, training data, input data, and/or hyper parameters (e.g., stride size in applying each convolutional kernel) for the system 200. The user may further receive outputs (e.g., the output 152) from the system 200 via the device 220.

[00037] FIG. 3 depicts a flowchart illustrating an example of a process 300 for training a convolutional neural network, in accordance with some example embodiments. Referring to FIGS. 1-3, the system 200 may perform the process 300 to train the convolutional neural network 100, which may have a tree feature map selection layer and/or a tree pooling layer.

[00038] At 302, the system 200 may receive training data. For example, the system 200 may receive training data directly from a user or from the device 220. In some example embodiments, training data may include at least one training input and a correct output corresponding to that training input.

[00039] At 304, the system 200 may utilize the training data to train a convolutional neural network having a tree feature map selection layer and/or a tree pooling layer. For example, the system 200 may train the convolutional neural network 100 by using the convolutional neural network 100 to process a plurality of training inputs. For each training input, an error associated with the training output of the convolutional neural network 100 (e.g., the output 152) relative to the corresponding correct output may be back propagated through the convolutional neural network 100 to generate a loss function. Gradient descent may be performed in order to determine one or more optimizations to the convolutional neural network 100 which would minimize the loss function. In some example embodiments, training the convolutional neural network 100 may include using the convolutional neural network 100 to process any appropriate or desired number of training inputs. As such, the system 200 may perform multiple iterations of optimizations (e.g., adjustments of weights, biases, and/or parameters) in order to generate a trained convolutional neural network 100.

[00040] In some example embodiments, soft decision trees may be applied at both the tree feature map selection layer and the tree pooling layer. For example, the first soft decision tree 132 may be applied at the tree feature map selection layer 130 of the convolutional neural network 100. Meanwhile, the second soft decision tree 142 may be applied at the tree pooling layer 140 of the convolutional neural network 100. The application of soft decision trees may enable training of the convolutional neural network 100 in a supervised learning mode including backpropagation and gradient descent.

[00041] A conventional decision tree makes “hard” decisions in accordance with the following splitting function, which provides a discrete, non-continuous selection between different responses according to the following:

$$s(t_m) \in \{0, 1\}$$

[00042] This type of splitting function provides a “hard” decision and is not differentiable. As such, a conventional decision tree is incompatible with a training paradigm that employs techniques such as backpropagation and gradient descent. By contrast, a soft decision tree makes “soft” decisions in accordance with splitting functions that provide a continuous and differentiable selection between different responses. As such, the convolutional neural network 100, which has at least one tree feature map selection layer or tree pooling layer, may be trained in a supervised learning mode including backpropagation and gradient descent.

[00043] At 306, a trained convolutional neural network having one or more of a tree feature map selection layer and tree pooling layer may be provided. For example, a trained convolutional neural network 100 may be deployed to process actual input data and provide an output (e.g., classification of the input data). In some example embodiments, the trained convolutional neural network may be provided in any appropriate or desired manner including computer software, dedicated circuitry (e.g., ASICs), and/or over a cloud platform.

[00044] The process 300 may include additional and/or different operations than shown without departing from the scope of the present disclosure. For example, one or more operations of the process 300 may be repeated and/or omitted without departing from the scope of the present disclosure.

[00045] FIG. 4 depicts a flowchart illustrating an example of a process 400 for training a convolutional neural network, in accordance with some example embodiments. Referring to FIGS. 1-2 and 4, in some example embodiments, the process 400 may be performed by the system 200 to train the convolutional neural network 100 and may implement operation 304 of the process 300.

[00046] At 402, the system 200 may generate a plurality of feature maps by applying one or more convolutional kernels to training input data. For example, the system 200 (e.g., the convolutional module 210) may apply the first convolutional kernel 122 to the training input data to generate the first feature map 126. The system 200 may also apply the second convolutional kernel 124 to the training input data to generate the second feature map 126. In some example embodiments, the system 200 may apply any number of convolutional kernels to the training input data to generate the plurality of feature maps.

[00047] At 404, the system 200 may generate at least one selected feature map by applying a soft decision tree to at least some of the plurality of feature maps. For example, the system 200 (e.g., the tree feature map selection module 212) may apply the first soft decision tree 132 to make a “soft” selection over the first feature map 126 and the second feature map 128. Specifically, the first soft decision tree 132 may combine the first feature map 126 and the second feature map 128 to generate the selected feature map 134. The soft decision tree 132 may apply a “soft” splitting function at a decision node of the first soft decision tree 132 that combines the first feature map 126 and the second feature map 128 according to a mixing proportion. The “soft” splitting function may be a sigmoid function that determines the mixing proportion based on the first feature map 126 and the second feature map 128. The mixing proportion may indicate a portion (e.g., percentage) of the first feature map 126 and a portion of the second feature map 128 to include in the selected feature map 134.

[00048] At 406, the system 200 may generate a pooled feature map by applying a soft decision tree to each portion of a feature map. For example, the system 200 (e.g., the tree pooling module 214) may apply the second soft decision tree 142 to the first portion 134A of the selected feature map 134 to generate the corresponding second portion 144A in the pooled

feature map 144. In some example embodiments, the pooled feature map 144 may provide a more robust and compact representation of the selected feature map 134.

[00049] Different pooling filters may be applied to the first portion 134 A. In some example embodiments, the second soft decision tree 142 may make a “soft” decision over the outputs from the different pooling filters. For example, the second soft decision tree 142 may apply a “soft” splitting function at a decision node to combine the outputs from the different pooling filter at each leaf node. The outputs from the different pooling filters may be combined according to a mixing proportion. The mixing proportion may indicate a portion (e.g., percentage) of the output from each pooling filter to include in the combination of the outputs. Meanwhile, the “soft” splitting function may be a sigmoid function that determines the mixing proportion based on the portion of the selected feature map 134 being pooled (e.g., the first portion 134A).

[00050] At 408, the system 200 may determine a training output based at least in part on one or more pooled feature maps. For example, the system 200 (e.g., the output module 216) may generate the output 152 based at least in part on the pooled feature map 144. In some example embodiments, the training out (e.g., the output 152) may exhibit an error relative to the correct output associated with the training input data.

[00051] At 410, the system 200 may determine one or more optimizations based at least in part on the training output. In some example embodiments, the system 200 may determine optimizations to the convolutional neural network 100 based on the error associated with the training output (e.g., the output 152). The optimizations may include adjustments to parameters applied at the tree feature selection map layer 130 and the tree pooling layer 140. Both the tree feature selection map 130 and the tree pooling layer 140 apply soft decision trees (e.g., the first

soft decision tree 132 and the second soft decision tree 142). As such, the system 200 may determine optimizations to the convolutional neural network 100 at both the tree feature selection map layer 130 and the tree pooling layer 140 using techniques such as backpropagation and gradient descent.

[00052] As shown in FIG. 1, the first soft decision tree 132 at the tree feature map selection layer 130 may receive a set of feature maps from the convolutional layer 120. The set of feature maps may be denoted as follows:

$$WX + B,$$

wherein $W \in \mathbb{R}^{N_{out} \times N_{in}}$ is the weight matrix of a convolutional kernel (e.g., the first convolutional kernel 122 or the second convolutional kernel 124) with input X (e.g., the input data 110) and biases B . The number of output channels is denoted by N_{out} and the number of input channels is denoted by N_{in} . The set of feature maps may be decomposed as follows:

$$WX + B = [W_1; W_2]X + [B_1; B_2], \text{ wherein } W_1; W_2 \in \mathbb{R}^{(N_{out}/2) \times N_{in}}$$

[00053] The first soft decision tree 132 may make “soft” selections over the set of feature maps $WX + B$ (e.g., to generate the selected feature map 134) according to the following splitting function:

$$f(X) = \mathbf{p}^\circ(W_1X + B_1) + (1 - \mathbf{p})^\circ(W_2X + B_2),$$

wherein $\mathbf{p}$ and $(1 - \mathbf{p})$ may represent the mixing proportion used at a decision node in the first soft decision tree 132 to combine responses from the child nodes of that decision node (e.g., the first feature map 126 and the second feature map 128), $\mathbf{1}$ may represent a vector where all elements are one, $^\circ$ denotes a Hadamard product, and $\mathbf{p}$ may be defined as follows:

$$\mathbf{p} = s(\mathbf{t}) \in \mathbb{R}^{N_{out}/2},$$

wherein $s(\mathbf{t})$ is a splitting function (e.g., a sigmoid function) that determines the mixing proportion $\mathbf{p}$.

[00054] Optimizations to the convolutional neural network 100 may include adjusting the weights W and biases B applied at the tree feature map selection layer 130 in order to minimize an error E associated with the training output (e.g., the output 152). Specifically, adjustments may be made according to the following partial derivatives of the splitting function $f(X)$ with respect to the weights W and biases B :

$$\frac{\partial E}{\partial W_1} = \mathbf{p}^\circ \partial X^T$$

$$\frac{\partial E}{\partial W_2} = (1 - \mathbf{p})^\circ \partial X^T$$

$$\frac{\partial E}{\partial B_1} = \mathbf{p}^\circ \partial$$

$$\frac{\partial E}{\partial B_2} = (1 - \mathbf{p})^\circ \partial$$

$$\frac{\partial E}{\partial \mathbf{t}} = \mathbf{p}^\circ (1 - \mathbf{p})^\circ \delta [(W_1 X + B_1) - (W_2 X + B_2)], \text{ wherein } \delta = \partial E / \partial f(X) \in \mathbb{R}^{\frac{N_{out}}{2}}.$$

[00055] Further backpropagation of the error E to a preceding layer (e.g., the convolutional layer 120) may be determined based on the following:

$$\frac{\partial E}{\partial X} = \delta^T [\mathbf{p}^\circ W_1 + (1 - \mathbf{p})^\circ W_2].$$

[00056] Similarly, a decision node (e.g., parent node, root node) in the second soft decision tree 142 at the tree pooling layer 140 may make “soft” decisions over outputs from that decision node’s child nodes. For example, a decision node may make a “soft” decision over the outputs from the different pooling filters (e.g., maximum pooling filter, average pooling filter, stochastic pooling filter, and/or the like) that are applied at the leaf nodes of the second soft

decision tree 142 to an individual portion (e.g., the first portion 134A) of a feature map (e.g., the selected feature map 134). As such, the output at each node m of the second soft decision tree 142 may be defined as follows:

$$f_m(x) = \begin{cases} \mathbf{w}_m^T \mathbf{x}, & \text{if leaf node} \\ s(t_m)f_{m,\text{left}}(x) + (1 - s(t_m))f_{m,\text{right}}(x), & \text{otherwise} \end{cases}$$

wherein $\mathbf{w}_m \in \mathbb{R}^N$ may be individual pooling filters and the splitting function $s(t_m)$ may be a sigmoid function that is applied at each decision node m based on the splitting parameter t_m . In some example embodiments, the splitting parameter t_m may be a weight that is applicable to an output at each decision node m . The splitting function $s(t_m)$ may be defined as follows:

$$s(t_m) = \frac{1}{1 + \exp\{-t_m\}} \in [0, 1].$$

[00057] Optimizations to the convolutional neural network 100 may include adjusting the pooling filters $\mathbf{w}_m$ and the splitting parameters t_m applied at the tree pooling layer 140 in order to minimize the error E associated with the output 152. For example, the following pooling function $f(x)$ denotes the output function of the second soft decision tree 142 where the second soft decision tree 142 has a single layer (e.g., a pair of leaf nodes descending directly from a root node):

$$f(x) = p\mathbf{w}_1^T \mathbf{x} + (1 - p)\mathbf{w}_2^T \mathbf{x},$$

wherein p and $(1 - p)$ may represent the mixing proportion used at the root node of the second soft decision tree 142 to combine outputs from the different pooling filter at each leaf node.

[00058] As such, adjustments may be made according to the following partial derivatives of the pooling function $f(x)$ with respect to the different pooling filters $\mathbf{w}_1$ and $\mathbf{w}_2$:

$$\frac{\partial E}{\partial \mathbf{w}_1} = \frac{\partial E}{\partial f(x)} \frac{\partial f(x)}{\partial \mathbf{w}_1} = \delta p \mathbf{x}$$

$$\frac{\partial E}{\partial \mathbf{w}_2} = \frac{\partial E}{\partial f(x)} \frac{\partial f(x)}{\partial \mathbf{w}_2} = \delta(1 - p) \mathbf{x}$$

$$\frac{\partial E}{\partial t} = \frac{\partial E}{\partial f(x)} \frac{\partial f(x)}{\partial t} = \delta p(1-p)(\mathbf{w}_1^T \mathbf{x} - \mathbf{w}_2^T \mathbf{x}),$$

wherein p may represent the value of the response from the splitting function $s(t)$ and $\mathbf{x}$ is the output response from a previous layer.

[00059] The error E to be further propagated from the tree pooling layer 140 to a preceding layer (e.g., the tree feature map selection layer 130) may be defined as follows:

$$\frac{\partial E}{\partial X} = \frac{\partial E}{\partial f(x)} \frac{\partial f(x)}{\partial t} = \delta[p\mathbf{w}_1 + (1-p)\mathbf{w}_2]$$

[00060] The process 400 may include additional and/or different operations than shown without departing from the scope of the present disclosure. For example, one or more operations included in the process 400 may be repeated and/or omitted without departing from the scope of the present disclosure. Moreover, when training a convolutional neural network, the process 400 may be repeated any appropriate or desired number of times (e.g., using different input data) to achieve an optimal convolutional neural network.

[00061] FIG. 5 depicts an example of a soft decision tree 500, in accordance with some embodiments. Referring to FIGS. 1-5, in some example embodiments, the soft decision tree 500 may implement the second soft decision tree 144.

[00062] The soft decision tree 500 may include a plurality of nodes. As shown in FIG. 5, the soft decision tree 500 may include a plurality of child nodes. Each child node is associated with an output from the application of a different pooling filter to the first portion 144A of the selected feature map 144 including a first pooling output 510, a second pooling output 512, a third pooling output 514, and a fourth pooling output 516. Each pooling filter may apply a subsampling operation (e.g., maximum, average, stochastic, and/or the like) on the first portion 144A. As such, each of the first pooling output 510, the second pooling output 512, the third

pooling output 514, and the fourth pooling output 516 may be a subsample of the first portion 144A of the selected feature map 144.

[00063] The soft decision tree 500 may further include a plurality of decision nodes including a first parent node 522, a second parent node 524, and a root node 526. Each decision node may apply a “soft” splitting function which combines the outputs from that decision node’s child nodes according to a mixing proportion. The mixing proportion may indicate a portion (e.g., percentage) of the outputs from each child node to include in an output from the parent node. In some example embodiments, the first parent node 522, the second parent node 524, and the root node 526 may each apply the following splitting function:

$$s(t_m) = \frac{1}{1 + \exp\{-t_m\}} \in [0, 1],$$

wherein $s(t_m)$ may be a sigmoid function that determines the mixing proportion at each node m of the soft decision tree 500 in accordance with the splitting parameter t_m .

[00064] As such, the output at each node of the soft decision tree 500 may be denoted as follows:

$$f_m(x) = \begin{cases} \mathbf{w}_m^T \mathbf{x}, & \text{if leaf node} \\ s(t_m)f_{m,\text{left}}(x) + (1 - s(t_m))f_{m,\text{right}}(x), & \text{otherwise} \end{cases}$$

wherein $\mathbf{w}_m \in \mathbb{R}^N$ are individual pooling filters applied to each portion of the selected feature map 134 (e.g., the first portion 134A).

[00065] In some example embodiments, the soft decision tree 500 may be part of a convolutional neural network (e.g., the convolutional neural network 100). For example, the soft decision tree 500 may be applied at a tree pooling layer (e.g., the tree pooling layer 140) of the convolutional neural network. As such, training the convolutional neural network may include adjusting the pooling filters $\mathbf{w}_m$ and the splitting parameters t_m of the soft decision tree 500 in

order to minimize the error E associated with an output of the convolutional neural network (e.g., the output 152). The adjustments may be made in a supervised learning mode where the error E is back propagated through the convolutional neural network to determine a loss function. Gradient descent may be performed to determine the pooling filters $\mathbf{w}_m$ and the splitting parameters t_m that would minimize the loss function.

[00066] Although the soft decision tree 500 is shown to include two levels of decision nodes, the soft decision tree 500 can include a different number of levels of decision nodes without departing from the scope of the present disclosure. For example, in some example embodiments, the soft decision tree 500 may include a single level of decision nodes.

[00067] FIG. 6 depicts a flowchart illustrating a process 600 for utilizing a trained convolutional neural network, in accordance with some example embodiments. Referring to FIGS. 1-2 and 6, in some example embodiments, the process 600 may be performed by the system 200 to utilize the convolutional neural network 100 subsequent to training.

[00068] At 602, the system 200 may receive input data. For example, the system 200 may receive input data directly from a user or from the device 220. In some example embodiments, the input data may be any type of data including image, speech, genomic data, and/or any other type of data.

[00069] At 604, the system 200 may process the input data by at least utilizing a trained convolutional neural network having at least one of a tree feature map selection layer and a tree pooling layer. For example, the system 200 may utilize the trained convolutional neural network 100 to process the input data. The trained convolutional neural network 100 may include at least one of the tree feature map selection layer 130 and the tree pooling layer 140.

[00070] At 606, the system 200 may provide, as an output, a result of the processing performed by the trained convolutional neural network. For example, the result of the processing performed by the trained convolutional neural network 100 may be a classification of the input data. In some example embodiments, the system 200 may provide the output directly to a user or via the device 220.

[00071] The process 600 may include additional and/or different operations than shown without departing from the scope of the present disclosure. For example, one or more operations of the process 600 may be repeated and/or omitted without departing from the scope of the present disclosure.

[00072] One or more aspects or features of the subject matter described herein can be realized in digital electronic circuitry, integrated circuitry, specially designed application specific integrated circuits (ASICs), field programmable gate arrays (FPGAs) computer hardware, firmware, software, and/or combinations thereof. These various aspects or features can include implementation in one or more computer programs that are executable and/or interpretable on a programmable system including at least one programmable processor, which can be special or general purpose, coupled to receive data and instructions from, and to transmit data and instructions to, a storage system, at least one input device, and at least one output device. The programmable system or computing system may include clients and servers. A client and server are generally remote from each other and typically interact through a communication network. The relationship of client and server arises by virtue of computer programs running on the respective computers and having a client-server relationship to each other.

[00073] These computer programs, which can also be referred to as programs, software, software applications, applications, components, or code, include machine instructions for a

programmable processor, and can be implemented in a high-level procedural and/or object-oriented programming language, and/or in assembly/machine language. As used herein, the term “machine-readable medium” refers to any computer program product, apparatus and/or device, such as for example magnetic discs, optical disks, memory, and Programmable Logic Devices (PLDs), used to provide machine instructions and/or data to a programmable processor, including a machine-readable medium that receives machine instructions as a machine-readable signal. The term “machine-readable signal” refers to any signal used to provide machine instructions and/or data to a programmable processor. The machine-readable medium can store such machine instructions non-transitorily, such as for example as would a non-transient solid-state memory or a magnetic hard drive or any equivalent storage medium. The machine-readable medium can alternatively or additionally store such machine instructions in a transient manner, such as for example, as would a processor cache or other random access memory associated with one or more physical processor cores.

[00074] To provide for interaction with a user, one or more aspects or features of the subject matter described herein can be implemented on a computer having a display device, such as for example a cathode ray tube (CRT) or a liquid crystal display (LCD) or a light emitting diode (LED) monitor for displaying information to the user and a keyboard and a pointing device, such as for example a mouse or a trackball, by which the user may provide input to the computer. Other kinds of devices can be used to provide for interaction with a user as well. For example, feedback provided to the user can be any form of sensory feedback, such as for example visual feedback, auditory feedback, or tactile feedback; and input from the user may be received in any form, including, but not limited to, acoustic, speech, or tactile input. Other possible input devices include, but are not limited to, touch screens or other touch-sensitive

devices such as single or multi-point resistive or capacitive track pads, voice recognition hardware and software, optical scanners, optical pointers, digital image capture devices and associated interpretation software, and the like.

[00075] The subject matter described herein can be embodied in systems, apparatus, methods, and/or articles depending on the desired configuration. The implementations set forth in the foregoing description do not represent all implementations consistent with the subject matter described herein. Instead, they are merely some examples consistent with aspects related to the described subject matter. Although a few variations have been described in detail above, other modifications or additions are possible. In particular, further features and/or variations can be provided in addition to those set forth herein. For example, the implementations described above can be directed to various combinations and subcombinations of the disclosed features and/or combinations and subcombinations of several further features disclosed above. In addition, the logic flows depicted in the accompanying figures and/or described herein do not necessarily require the particular order shown, or sequential order, to achieve desirable results. Other implementations may be within the scope of the following claims.

WHAT IS CLAIMED IS:

1. A method comprising:
receiving training data;
utilizing the training data to train a convolutional neural network comprising a tree pooling layer, wherein the tree pooling layer applies a soft decision tree to generate one or more pooled feature map; and
providing a trained convolutional neural network comprising a tree pooling layer.
2. The method as recited in claim 1, wherein the convolutional neural network comprises a convolutional layer configured to generate a plurality of feature maps based at least in part on the training data.
3. The method as recited in claim 2, wherein the convolutional layer generates at least one feature map by at least applying a convolutional kernel to the training data.
4. The method as recited in claim 3, wherein the convolutional kernel is adapted to detect a feature in the training data.
5. The method as recited in claim 4, wherein the at least one feature map depicts one or more occurrences of the feature in the training data.
6. The method as recited in claim 2, wherein the convolutional neural network further comprises a tree feature map selection layer configured to generate at least one selected feature map based at least in part on the plurality of feature maps generated at the convolutional layer.

7. The method as recited in claim 6, wherein the tree pooling layer is configured to apply the soft decision tree to each portion of the selected feature map to generate a corresponding portion of the pooled feature map.
8. The method as recited in claim 7, wherein the soft decision tree comprises a plurality of leaf nodes and decision nodes, wherein each leaf node corresponds to a pooling filter to apply to a portion of the selected feature map, and wherein a decision node applies a soft splitting function that combines an output from each child node of that decision node according to a mixing proportion.
9. The method as recited in claim 8, wherein the pooling filter comprises one of a maximum operation, an average operation, and a stochastic operation.
10. The method as recited in claim 8, wherein the mixing proportion indicates a portion of the output from each child node to include in a combination of the outputs from the child nodes.
11. The method as recited in claim 6, wherein the tree feature map selection layer is configured to apply a soft decision tree to generate the at least one selected feature map.
12. The method as recited in claim 11, wherein the soft decision tree combines two or more of the plurality of the feature maps into the at least one selected feature map.
13. The method as recited in claim 12, wherein the soft decision tree combines the two or more feature maps according to a mixing proportion.
14. The method as recited in claim 13, wherein the mixing proportion indicates a portion of each of the two or more feature maps to include in the selected feature map.

15. The method as recited in claim 1, wherein the convolutional neural network further comprises an output layer configured to generate a training output based at least in part on the one or more pooled feature maps.

16. The method as recited in claim 15, wherein training the convolutional neural network includes determining, by at least backpropagation and gradient descent, one or more optimizations based at least in part on an error associated with the training output.

17. The method as recited in claim 1, wherein providing the trained convolutional neural network comprises one of sending and/or storing the trained convolutional neural network.

18. A method comprising:

receiving input data;

processing the input data by at least utilizing a trained convolutional neural network comprising a tree pooling layer, wherein the tree pooling layer applies a soft decision tree to generate one or more pooled feature maps; and

providing, as an output, a result of the processing performed by the trained convolutional neural network.

19. The method as recited in claim 18, wherein the trained convolutional neural network further comprises a convolutional layer, wherein the convolutional layer is configured to generate a plurality of feature maps based at least in part on the input data, and wherein the convolutional layer generates each of the plurality of feature maps by at least applying a convolutional kernel to the input data.

20. The method as recited in claim 19, wherein the trained convolutional neural network further comprises a tree feature map selection layer, wherein the tree feature map selection layer is configured to apply a soft decision tree to generate at least one selected feature map, and wherein the soft decision tree generates the at least one feature map by at least combining two or more of the plurality of feature maps generated at the convolutional layer.

21. A system comprising:

at least one processor; and

at least one memory including program code which when executed by the at least one memory provides operations comprising

receiving training data;

utilizing the training data to train a convolutional neural network comprising a tree pooling layer, wherein the tree pooling layer applies a soft decision tree to generate one or more pooled feature map; and

providing a trained convolutional neural network comprising a tree pooling layer.

22. The system as recited in claim 21, wherein the convolutional neural network comprises a convolutional layer configured to generate a plurality of feature maps based at least in part on the training data.

23. The system as recited in claim 22, wherein the convolutional layer generates at least one feature map by at least applying a convolutional kernel to the training data.

24. The system as recited in claim 23, wherein the convolutional kernel is adapted to detect a feature in the training data.

25. The system as recited in claim 24, wherein the at least one feature map depicts one or more occurrences of the feature in the training data.

26. The system as recited in claim 22, wherein the convolutional neural network further comprises a tree feature map selection layer configured to generate at least one selected feature map based at least in part on the plurality of feature maps generated at the convolutional layer.

27. The system as recited in claim 26, wherein the tree pooling layer is configured to apply the soft decision tree to each portion of the selected feature map to generate a corresponding portion of the pooled feature map.

28. The system as recited in claim 27, wherein the soft decision tree comprises a plurality of leaf nodes and decision nodes, wherein each leaf node corresponds to a pooling filter to apply to a portion of the selected feature map, and wherein a decision node applies a soft splitting function that combines an output from each child node of that decision node according to a mixing proportion.

29. The system as recited in claim 28, wherein the pooling filter comprises one of a maximum operation, an average operation, and a stochastic operation.

30. The system as recited in claim 28, wherein the mixing proportion indicates a portion of the output from each child node to include in a combination of the outputs from the child nodes.

31. The system as recited in claim 26, wherein the tree feature map selection layer is configured to apply a soft decision tree to generate the at least one selected feature map.

32. The system as recited in claim 31, wherein the soft decision tree combines two or more of the plurality of the feature maps into the at least one selected feature map.

33. The system as recited in claim 32, wherein the soft decision tree combines the two or more feature maps according to a mixing proportion.

34. The system as recited in claim 33, wherein the mixing proportion indicates a portion of each of the two or more feature maps to include in the selected feature map.

35. The system as recited in claim 21, wherein the convolutional neural network further comprises an output layer configured to generate a training output based at least in part on the one or more pooled feature maps.

36. The system as recited in claim 35, wherein training the convolutional neural network includes determining, by at least backpropagation and gradient descent, one or more optimizations based at least in part on an error associated with the training output.

37. The system as recited in claim 21, wherein providing the trained convolutional neural network comprises one of sending and/or storing the trained convolutional neural network.

38. A system comprising:

at least one processor; and

at least one memory including program code which when executed by the at least one memory provides operations comprising:

receiving input data;

processing the input data by at least utilizing a trained convolutional neural network comprising a tree pooling layer, wherein the tree pooling layer applies a soft decision tree to generate one or more pooled feature maps; and

providing, as an output, a result of the processing performed by the trained convolutional neural network.

39. The system as recited in claim 18, wherein the trained convolutional neural network further comprises a convolutional layer, wherein the convolutional layer is configured to generate a plurality of feature maps based at least in part on the input data, and wherein the convolutional layer generates each of the plurality of feature maps by at least applying a convolutional kernel to the input data.

40. The system as recited in claim 39, wherein the trained convolutional neural network further comprises a tree feature map selection layer, wherein the tree feature map selection layer is configured to apply a soft decision tree to generate at least one selected feature map, and wherein the soft decision tree generates the at least one feature map by at least combining two or more of the plurality of feature maps generated at the convolutional layer.

41. A non-transitory computer-readable storage medium including program code which when executed by at least one processor causes operations comprising:

receiving training data;

utilizing the training data to train a convolutional neural network comprising a tree pooling layer, wherein the tree pooling layer applies a soft decision tree to generate one or more pooled feature map; and

providing a trained convolutional neural network comprising a tree pooling layer.

42. A non-transitory computer-readable storage medium including program code which when executed by at least one processor causes operations comprising:

receiving input data;

processing the input data by at least utilizing a trained convolutional neural network comprising a tree pooling layer, wherein the tree pooling layer applies a soft decision tree to generate one or more pooled feature maps; and

providing, as an output, a result of the processing performed by the trained convolutional neural network.

43. An apparatus comprising:

means for receiving training data;

means for utilizing the training data to train a convolutional neural network comprising a tree pooling layer, wherein the tree pooling layer applies a soft decision tree to generate one or more pooled feature map; and

means for providing a trained convolutional neural network comprising a tree pooling layer.

44. The apparatus as recited in claim 43, further comprising means for performing the method as recited in any of claims 2-17.

45. An apparatus comprising:

means for receiving input data;

means for processing the input data by at least utilizing a trained convolutional neural network comprising a tree pooling layer, wherein the tree pooling layer applies a soft decision tree to generate one or more pooled feature maps; and

means for providing, as an output, a result of the processing performed by the trained convolutional neural network.

46. The apparatus as recited in claim 45, further comprising means for performing the method as recited in any of claims 19-20.

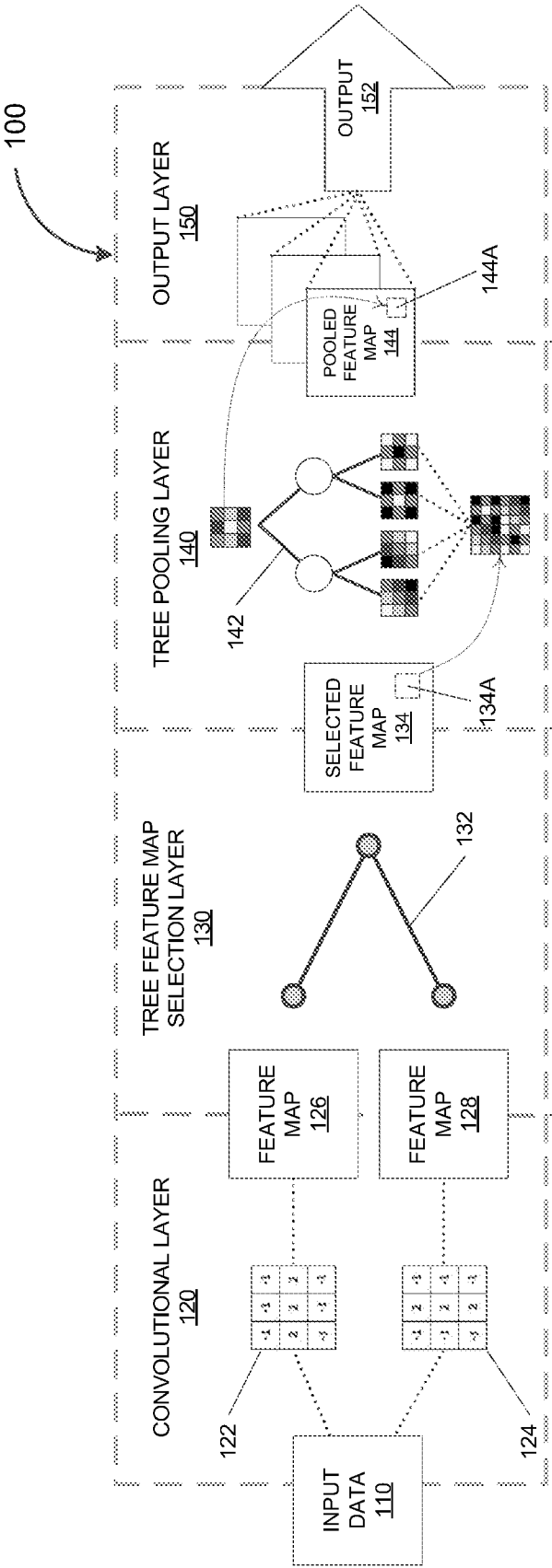
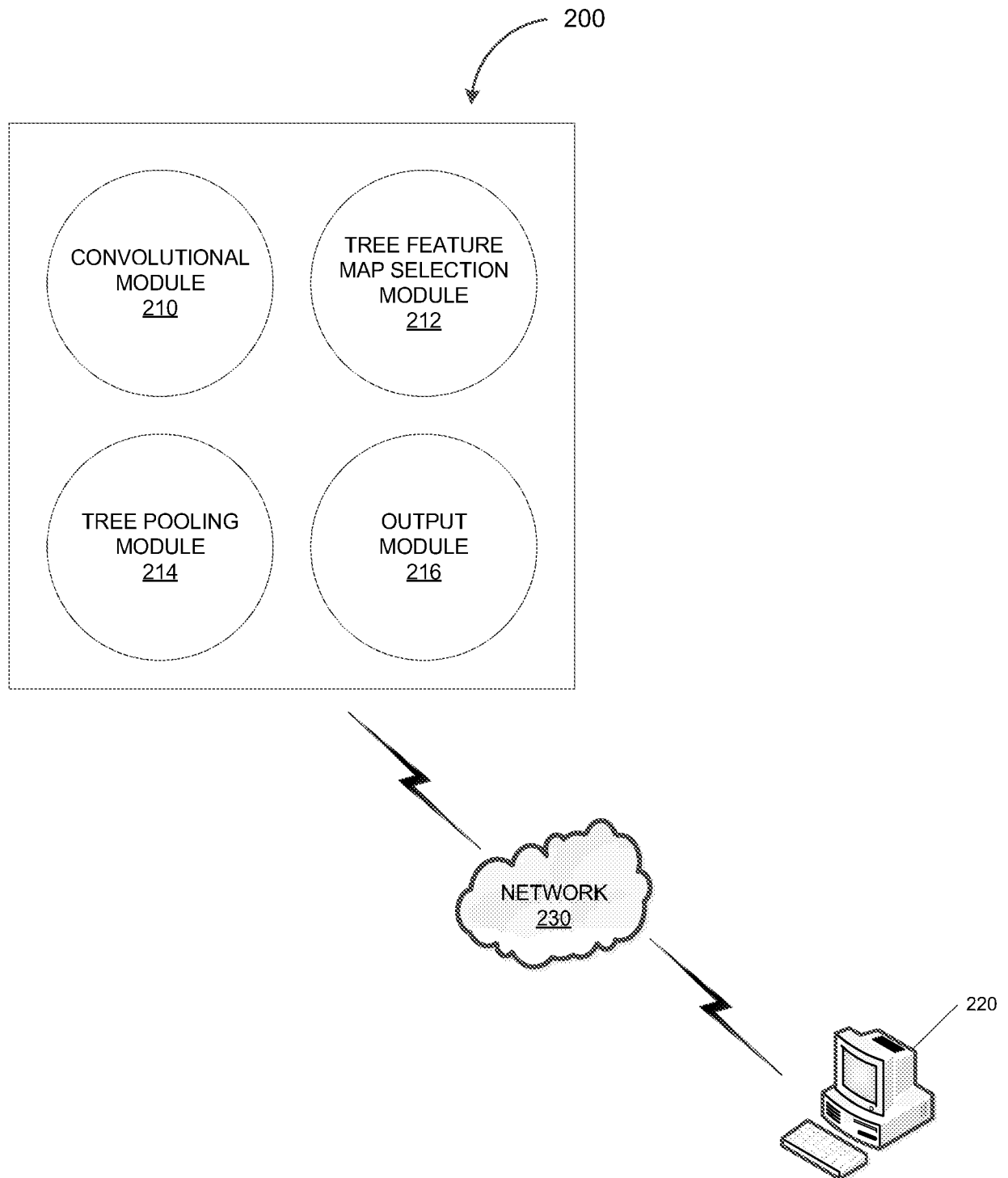
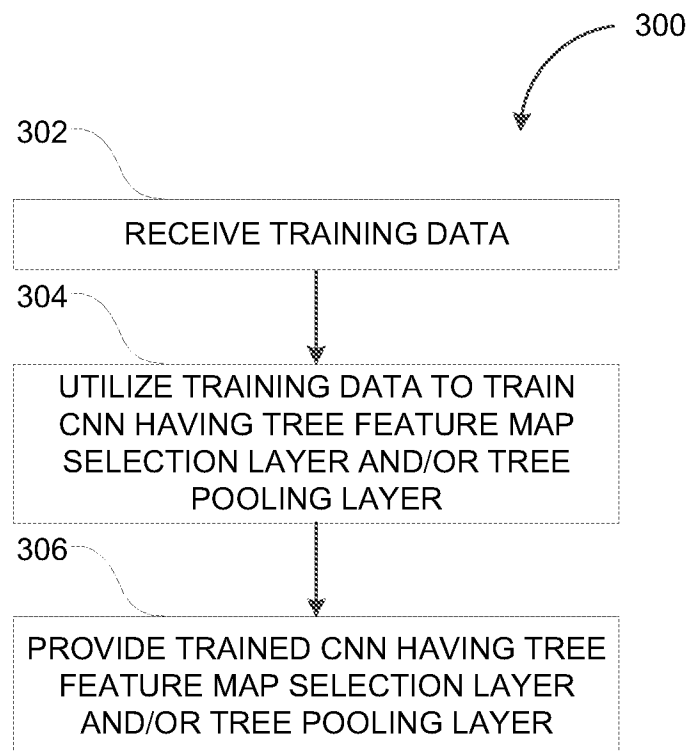
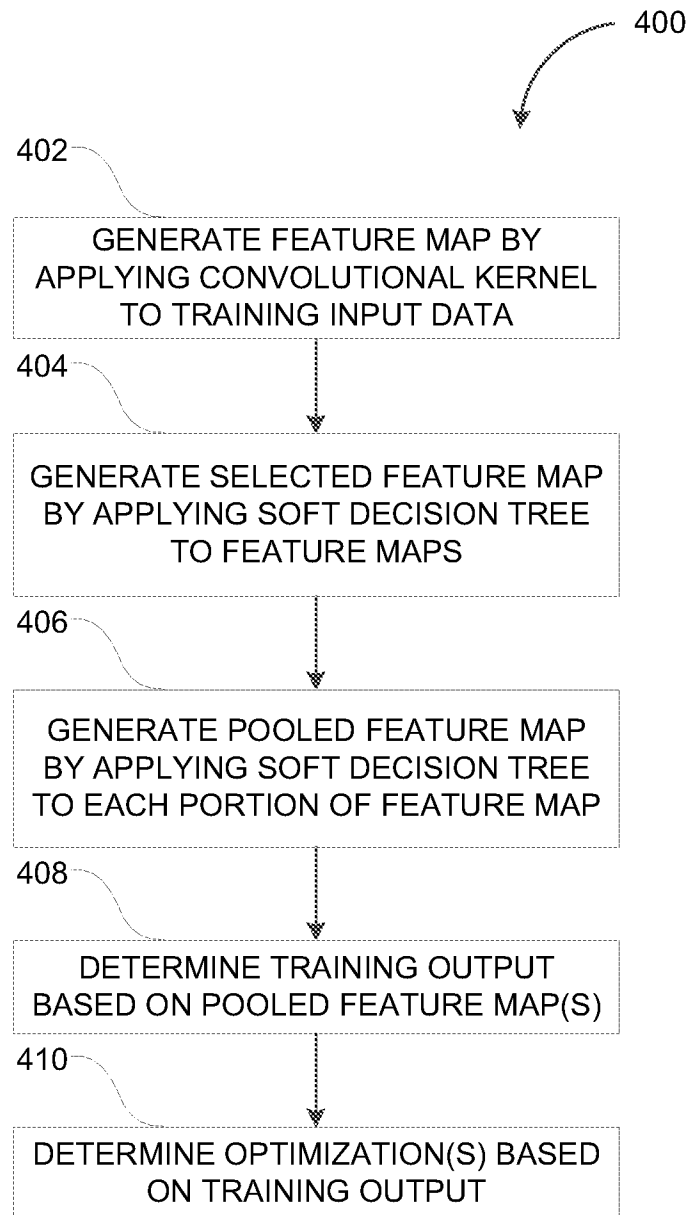


FIG. 1

2/6

**FIG. 2**

**FIG. 3**

**FIG. 4**

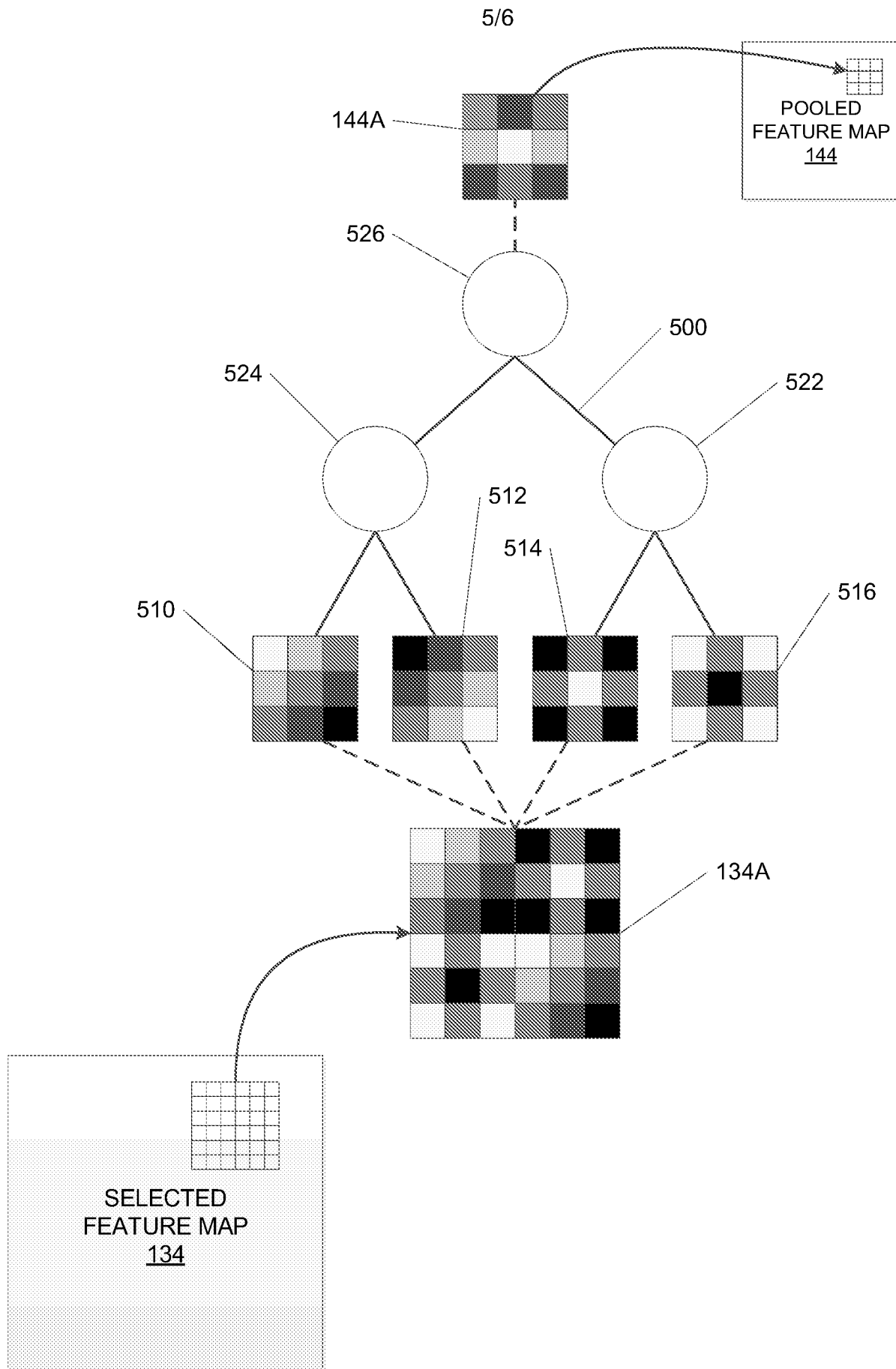
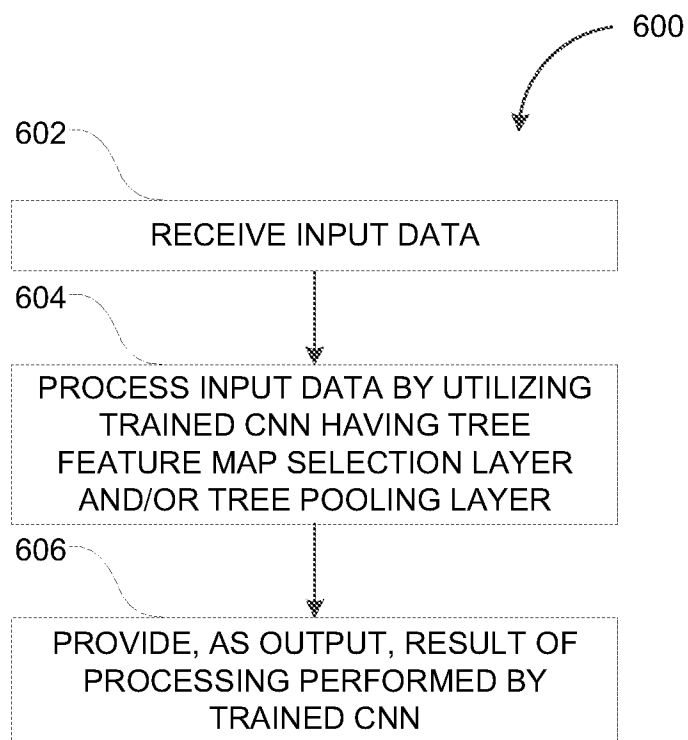


FIG. 5

**FIG. 6**

INTERNATIONAL SEARCH REPORT

International application No.

PCT/US2016/020869

A. CLASSIFICATION OF SUBJECT MATTER

IPC(8) - G06K 9/00; G06K 9/62; G06T 17/00; G10L 15/26 (2016.01)

CPC - G06T 2210/56; G06K 9/4628; G06K 9/6267; G06K 9/6272; G06N 3/0454; G06N 3/0481 (2016.01)

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

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)

IPC(8) - G06K 9/00, G06K 9/62, G06T 17/00, G10L 15/26 (2016.01)

CPC - G06K 9/4628, G06K 9/6267, G06K 9/6272, G06N 3/0454, G06N 3/0481, G06T 15/00, G06T 17/00, G06T 2210/56

G10L 15/16, G10L 15/26 (2016.01)

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

USPC - 382/103, 382/118, 382/156, 382/224, 704/235 (keyword delimited)

Electronic data base consulted during the international search (name of data base and, where practicable, search terms used)

Orbit, Google Patents, Google Scholar, Google

Search terms used: training data, train convolutional neural network, receive, tree pooling layer,

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
Y	US 2014/0270488 A1 (GOOGLE INC.) 18 September 2014 (18.09.2014), entire document	1-46
Y	US 2003/0002731 A1 (WERSING et al) 02 January 2003 (02.01.2003), entire document	1-46
Y	US 2015/0032449 A1 (NUANCE COMMUNICATIONS, INC.) 29 January 2015 (29.01.2015), entire document	1-17, 21-37, 41, 43, 44
Y	US 2015/0006444 A1 (DENSO CORPORATION) 01 January 2015 (01.01.2015), entire document	2-16, 19, 20, 22-36, 39, 40
Y	US 2009/0016470 A1 (VAN DER LAAN) 15 January 2009 (15.01.2009), entire document	6-14, 20, 26-34, 40
Y	US 2002/0083424 A1 (PASSERA et al) 27 June 2002 (27.06.2002), entire document	8-10, 28-30
A	WO 2011/088497 A1 (BAXTER) 28 July 2011 (28.07.2011), entire document	1-46
A	US 2015/0036920 A1 (FUJITSU LIMITED) 05 February 2015 (05.02.2015), entire document	1-46
A	US 7,912,246 B1 (MOON et al) 22 March 2011 (22.03.2011), entire document	1-46
A	US 2014/0288928 A1 (PENN et al) 25 September 2014 (25.09.2014), entire document	1-46
A	EP 2 418 643 A1 (SOFTWARE AG) 15 February 2012 (15.02.2012), entire document	1-46



Further documents are listed in the continuation of Box C.



See patent family annex.

* Special categories of cited documents:

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

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

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

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

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

"I"

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

"X"

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

"Y"

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

"&"

document member of the same patent family

Date of the actual completion of the international search

27 April 2016

Date of mailing of the international search report

09 JUN 2016

Name and mailing address of the ISA/

Mail Stop PCT, Attn: ISA/US, Commissioner for Patents

P.O. Box 1450, Alexandria, VA 22313-1450

Facsimile No. 571-273-8300

Authorized officer

Blaine R. Copenheaver

PCT Helpdesk: 571-272-4300

PCT OSP: 571-272-7774