

**TILE LOCATION AND/OR CYCLE BASED WEIGHT SET SELECTION
FOR BASE CALLING**

PRIORITY APPLICATIONS

[0001] This application claims the benefit of U.S. Provisional Patent Application No.: 63/161,880, titled “Tile Location and/or Cycle Based Weight Set Selection for Base Calling,” filed March 16, 2021 (Attorney Docket No. ILLM 1019-1/IP-1861-PRV); U.S. Provisional Patent Application No.: 63/161,896, titled “Neural Network Parameter Quantization for Base Calling,” filed March 16, 2021 (Attorney Docket No. ILLM 1019-2/IP-2049-PRV); U.S. Nonprovisional Patent Application No.: 17/687,551, titled “Tile Location and/or Cycle Based Weight Set Selection for Base Calling,” filed March 4, 2022 (Attorney Docket No. ILLM 1019-3/IP-1861-US); U.S. Nonprovisional Patent Application No.: 17,687,583, titled “Neural Network Parameter Quantization for Base Calling,” filed March 4, 2022 (Attorney Docket No. ILLM 1019-4/IP-2049-US). The priority applications are hereby incorporated by reference for all purposes.

FIELD OF THE TECHNOLOGY DISCLOSED

[0002] The technology disclosed relates to artificial intelligence type computers and digital data processing systems and corresponding data processing methods and products for emulation of intelligence (*i.e.*, knowledge based systems, reasoning systems, and knowledge acquisition systems); and including systems for reasoning with uncertainty (*e.g.*, fuzzy logic systems), adaptive systems, machine learning systems, and artificial neural networks. In particular, the technology disclosed relates to using deep neural networks such as deep convolutional neural networks for analyzing data, and selective use of weight sets.

INCORPORATIONS

[0003] The following are incorporated by reference as if fully set forth herein:

[0004] U.S. Provisional Patent Application No. 62/979,384, titled “ARTIFICIAL INTELLIGENCE-BASED BASE CALLING OF INDEX SEQUENCES,” filed 20 February 2020 (Attorney Docket No. ILLM 1015-1/IP-1857-PRV);

[0005] U.S. Provisional Patent Application No. 62/979,414, titled “ARTIFICIAL INTELLIGENCE-BASED MANY-TO-MANY BASE CALLING,” filed 20 February 2020 (Attorney Docket No. ILLM 1016-1/IP-1858-PRV);

[0006] U.S. Provisional Patent Application No. 62/979,385, titled “KNOWLEDGE DISTILLATION-BASED COMPRESSION OF ARTIFICIAL INTELLIGENCE-BASED BASE CALLER,” filed 20 February 2020 (Attorney Docket No. ILLM 1017-1/IP-1859-PRV);

- [0007] U.S. Provisional Patent Application No. 63/072,032, titled “DETECTING AND FILTERING CLUSTERS BASED ON ARTIFICIAL INTELLIGENCE-PREDICTED BASE CALLS,” filed 28 August 2020 (Attorney Docket No. ILLM 1018-1/IP-1860-PRV);
- [0008] U.S. Provisional Patent Application No. 62/979,411, titled “DATA COMPRESSION FOR ARTIFICIAL INTELLIGENCE-BASED BASE CALLING,” filed 20 February 2020 (Attorney Docket No. ILLM 1029-1/IP-1964-PRV);
- [0009] U.S. Provisional Patent Application No. 62/979,399, titled “SQUEEZING LAYER FOR ARTIFICIAL INTELLIGENCE-BASED BASE CALLING,” filed 20 February 2020 (Attorney Docket No. ILLM 1030-1/IP-1982-PRV);
- [0010] U.S. Nonprovisional Patent Application No. 16/825,987, titled “TRAINING DATA GENERATION FOR ARTIFICIAL INTELLIGENCE-BASED SEQUENCING,” filed 20 March 2020 (Attorney Docket No. ILLM 1008-16/IP-1693-US);
- [0011] U.S. Nonprovisional Patent Application No. 16/825,991 titled “ARTIFICIAL INTELLIGENCE-BASED GENERATION OF SEQUENCING METADATA,” filed 20 March 2020 (Attorney Docket No. ILLM 1008-17/IP-1741-US);
- [0012] U.S. Nonprovisional Patent Application No. 16/826,126, titled “ARTIFICIAL INTELLIGENCE-BASED BASE CALLING,” filed 20 March 2020 (Attorney Docket No. ILLM 1008-18/IP-1744-US);
- [0013] U.S. Nonprovisional Patent Application No. 16/826,134, titled “ARTIFICIAL INTELLIGENCE-BASED QUALITY SCORING,” filed 20 March 2020 (Attorney Docket No. ILLM 1008-19/IP-1747-US);
- [0014] U.S. Nonprovisional Patent Application No. 16/826,168, titled “ARTIFICIAL INTELLIGENCE-BASED SEQUENCING,” filed 21 March 2020 (Attorney Docket No. ILLM 1008-20/IP-1752-US);
- [0015] U.S. Nonprovisional Patent Application No. 16/874,599, titled “Systems and Devices for Characterization and Performance Analysis of Pixel-Based Sequencing,” filed May 14, 2020 (Attorney Docket No. ILLM 1011-4/IP-1750-US); and
- [0016] U.S. Nonprovisional Patent Application No. 17/176,147, titled “HARDWARE EXECUTION AND ACCELERATION OF ARTIFICIAL INTELLIGENCE-BASED BASE CALLER,” filed February 15, 2021 (Attorney Docket No. ILLM 1020-2/IP-1866-US).

BACKGROUND

[0017] The subject matter discussed in this section should not be assumed to be prior art merely as a result of its mention in this section. Similarly, a problem mentioned in this section or associated with the subject matter provided as background should not be assumed to have been

previously recognized in the prior art. The subject matter in this section merely represents different approaches, which in and of themselves can also correspond to implementations of the claimed technology.

[0018] The rapid improvement in computation capability has made deep Convolution Neural Networks (CNNs) a great success in recent years on many computer vision tasks with significantly improved accuracy. During the inference phase, many applications demand low latency processing of one image with strict power consumption requirement, which reduces the efficiency of Graphics Processing Unit (GPU) and other general-purpose platform, bringing opportunities for specific acceleration hardware, *e.g.*, Field Programmable Gate Array (FPGA), by customizing the digital circuit specific for the deep learning algorithm inference. However, deploying CNNs on portable and embedded systems is still challenging due to large data volume, intensive computation, varying algorithm structures, and frequent memory accesses.

[0019] As convolution contributes most operations in CNNs, the convolution acceleration scheme significantly affects the efficiency and performance of a hardware CNN accelerator. Convolution involves multiply and accumulate (MAC) operations with four levels of loops that slide along kernel and feature maps. The first loop level computes the MAC of pixels within a kernel window. The second loop level accumulates the sum of products of the MAC across different input feature maps. After finishing the first and second loop levels, a final output pixel is obtained by adding the bias. The third loop level slides the kernel window within an input feature map. The fourth loop level generates different output feature maps.

[0020] FPGAs have gained increasing interests and popularity in particular to accelerate the inference tasks, due to their (1) high degree of reconfigurability, (2) faster development time compared to application specific integrated circuits (ASICs) to catch up with the rapid evolving of CNNs, (3) good performance, and (4) superior energy efficiency compared to GPUs. The high performance and efficiency of an FPGA can be realized by synthesizing a circuit that is customized for a specific computation to directly process billions of operations with the customized memory systems. For instance, hundreds to thousands of Digital Signal Processing (DSP) blocks on modern FPGAs support the core convolution operation, *e.g.*, multiplication and addition, with high parallelism. Dedicated data buffers between external on-chip memory and on-chip Processing Engines (PEs) can be designed to realize the preferred dataflow by configuring tens of Mbyte on-chip Block Random Access Memories (BRAM) on the FPGA chip.

[0021] Efficient dataflow and hardware architecture of CNN acceleration are desired to minimize data communication while maximizing resource utilization to achieve high performance. An opportunity arises to design methodology and framework to accelerate the

inference process of various CNN algorithms on acceleration hardware with high performance, efficiency, and flexibility.

BRIEF DESCRIPTION OF THE DRAWINGS

[0022] In the drawings, like reference characters generally refer to like parts throughout the different views. Also, the drawings are not necessarily to scale, with an emphasis instead generally being placed upon illustrating the principles of the technology disclosed. In the following description, various implementations of the technology disclosed are described with reference to the following drawings, in which:

[0023] Fig. 1 illustrates a cross-section of a biosensor that can be used in various embodiments.

[0024] Fig. 2 depicts one implementation of a flow cell that contains clusters in its tiles.

[0025] Fig. 3 illustrates an example flow cell with eight lanes, and also illustrates a zoom-in on one tile and its clusters and their surrounding background.

[0026] Fig. 4 is a simplified block diagram of the system for analysis of sensor data from a sequencing system, such as base call sensor outputs.

[0027] Fig. 5 is a simplified diagram showing aspects of the base calling operation, including functions of a runtime program executed by a host processor.

[0028] Fig. 6 is a simplified diagram of a configuration of a configurable processor such as that of Fig. 4.

[0029] Fig. 7 is a diagram of a neural network architecture which can be executed using a configurable or a reconfigurable array configured as described herein.

[0030] Fig. 8A is a simplified illustration of an organization of tiles of sensor data used by a neural network architecture like that of Fig. 7.

[0031] Fig. 8B is a simplified illustration of patches of tiles of sensor data used by a neural network architecture like that of Fig. 7.

[0032] Fig. 9 illustrates part of a configuration for a neural network like that of Fig. 7 on a configurable or a reconfigurable array, such as a Field Programmable Gate Array (FPGA).

[0033] Fig. 10 is a diagram of another alternative neural network architecture which can be executed using a configurable or a reconfigurable array configured as described herein.

[0034] Fig. 11 illustrates one implementation of a specialized architecture of the neural network-based base caller that is used to segregate processing of data for different sequencing cycles.

[0035] Fig. 12 depicts one implementation of segregated layers, each of which can include convolutions.

- [0036] Fig. 13A depicts one implementation of combinatory layers, each of which can include convolutions.
- [0037] Fig. 13B depicts another implementation of the combinatory layers, each of which can include convolutions.
- [0038] Figs. 14, 15, and 16 illustrate various example tile-location based weight selection schemes used for base calling.
- [0039] Fig. 17A illustrates an example of fading, in which signal intensity is decreased as a function of cycle number is a sequencing run of a base calling operation.
- [0040] Fig. 17B conceptually illustrates a decreasing signal-to-noise ratio as cycles of sequencing progress.
- [0041] Fig. 18 illustrates an example base calling cycle-number based weight selection scheme used for base calling.
- [0042] Figs. 19, 20, 21A, and 21B illustrate various example weight selection schemes based on (i) temporal progression of base calling cycle numbers and (ii) spatial locations of tiles.
- [0043] Fig. 22 illustrates one implementation of base calling operations, in which weight sets used for the base calling are selected based on spatial tile information and temporal subseries of sensing cycle information.
- [0044] Fig. 23A illustrates various weights sets for various categories of tiles and for various sensing cycles, individual weight set comprising corresponding spatial weights and corresponding temporal weights.
- [0045] Fig. 23B illustrates various weights sets for various categories of tiles and for various cycles, wherein different weight sets of a specific category of tile includes common spatial weights and different temporal weights.
- [0046] Fig. 23C illustrates a system to select weight sets based on one or more sequencing run parameters.
- [0047] Fig. 24 is a block diagram of a base calling system in accordance with one implementation.
- [0048] Fig. 25 is a block diagram of a system controller that can be used in the system of Fig. 24.
- [0049] Fig. 26 is a simplified block diagram of a computer system that can be used to implement the technology disclosed.

DETAILED DESCRIPTION

- [0050] Embodiments described herein may be used in various biological or chemical processes and systems for academic or commercial analysis. More specifically, embodiments

described herein may be used in various processes and systems where it is desired to detect an event, property, quality, or characteristic that is indicative of a desired reaction. For example, embodiments described herein include cartridges, biosensors, and their components as well as bioassay systems that operate with cartridges and biosensors. In particular embodiments, the cartridges and biosensors include a flow cell and one or more sensors, pixels, light detectors, or photodiodes that are coupled together in a substantially unitary structure.

[0051] The following detailed description of certain embodiments will be better understood when read in conjunction with the appended drawings. To the extent that the figures illustrate diagrams of the functional blocks of various embodiments, the functional blocks are not necessarily indicative of the division between hardware circuitry. Thus, for example, one or more of the functional blocks (*e.g.*, processors or memories) may be implemented in a single piece of hardware (*e.g.*, a general purpose signal processor or random access memory, hard disk, or the like). Similarly, the programs may be standalone programs, may be incorporated as subroutines in an operating system, may be functions in an installed software package, and the like. It should be understood that the various embodiments are not limited to the arrangements and instrumentality shown in the drawings.

[0052] As used herein, an element or step recited in the singular and proceeded with the word “a” or “an” should be understood as not excluding plural of said elements or steps, unless such exclusion is explicitly stated. Furthermore, references to “one embodiment” are not intended to be interpreted as excluding the existence of additional embodiments that also incorporate the recited features. Moreover, unless explicitly stated to the contrary, embodiments “comprising” or “having” or “including” an element or a plurality of elements having a particular property may include additional elements whether or not they have that property.

[0053] As used herein, a “desired reaction” includes a change in at least one of a chemical, electrical, physical, or optical property (or quality) of an analyte-of-interest. In particular embodiments, the desired reaction is a positive binding event (*e.g.*, incorporation of a fluorescently labeled biomolecule with the analyte-of-interest). More generally, the desired reaction may be a chemical transformation, chemical change, or chemical interaction. The desired reaction may also be a change in electrical properties. For example, the desired reaction may be a change in ion concentration within a solution. Exemplary reactions include, but are not limited to, chemical reactions such as reduction, oxidation, addition, elimination, rearrangement, esterification, amidation, etherification, cyclization, or substitution; binding interactions in which a first chemical binds to a second chemical; dissociation reactions in which two or more chemicals detach from each other; fluorescence; luminescence; bioluminescence; chemiluminescence; and biological reactions, such as nucleic acid replication, nucleic acid

amplification, nucleic acid hybridization, nucleic acid ligation, phosphorylation, enzymatic catalysis, receptor binding, or ligand binding. The desired reaction can also be an addition or elimination of a proton, for example, detectable as a change in pH of a surrounding solution or environment. An additional desired reaction can be detecting the flow of ions across a membrane (*e.g.*, natural or synthetic bilayer membrane), for example as ions flow through a membrane the current is disrupted and the disruption can be detected.

[0054] In particular embodiments, the desired reaction includes the incorporation of a fluorescently-labeled molecule to an analyte. The analyte may be an oligonucleotide and the fluorescently-labeled molecule may be a nucleotide. The desired reaction may be detected when an excitation light is directed toward the oligonucleotide having the labeled nucleotide, and the fluorophore emits a detectable fluorescent signal. In alternative embodiments, the detected fluorescence is a result of chemiluminescence or bioluminescence. A desired reaction may also increase fluorescence (or Förster) resonance energy transfer (FRET), for example, by bringing a donor fluorophore in proximity to an acceptor fluorophore, decrease FRET by separating donor and acceptor fluorophores, increase fluorescence by separating a quencher from a fluorophore or decrease fluorescence by co-locating a quencher and fluorophore.

[0055] As used herein, a “reaction component” or “reactant” includes any substance that may be used to obtain a desired reaction. For example, reaction components include reagents, enzymes, samples, other biomolecules, and buffer solutions. The reaction components are typically delivered to a reaction site in a solution and/or immobilized at a reaction site. The reaction components may interact directly or indirectly with another substance, such as the analyte-of-interest.

[0056] As used herein, the term “reaction site” is a localized region where a desired reaction may occur. A reaction site may include support surfaces of a substrate where a substance may be immobilized thereon. For example, a reaction site may include a substantially planar surface in a channel of a flow cell that has a colony of nucleic acids thereon. Typically, but not always, the nucleic acids in the colony have the same sequence, being for example, clonal copies of a single stranded or double stranded template. However, in some embodiments a reaction site may contain only a single nucleic acid molecule, for example, in a single stranded or double stranded form. Furthermore, a plurality of reaction sites may be unevenly distributed along the support surface or arranged in a predetermined manner (*e.g.*, side-by-side in a matrix, such as in microarrays). A reaction site can also include a reaction chamber (or well) that at least partially defines a spatial region or volume configured to compartmentalize the desired reaction.

[0057] This application uses the terms “reaction chamber” and “well” interchangeably. As used herein, the term “reaction chamber” or “well” includes a spatial region that is in fluid

communication with a flow channel. The reaction chamber may be at least partially separated from the surrounding environment or other spatial regions. For example, a plurality of reaction chambers may be separated from each other by shared walls. As a more specific example, the reaction chamber may include a cavity defined by interior surfaces of a well and have an opening or aperture so that the cavity may be in fluid communication with a flow channel. Biosensors including such reaction chambers are described in greater detail in international application no. PCT/US2011/057111, filed on October 20, 2011, which is incorporated herein by reference in its entirety.

[0058] In some embodiments, the reaction chambers are sized and shaped relative to solids (including semi-solids) so that the solids may be inserted, fully or partially, therein. For example, the reaction chamber may be sized and shaped to accommodate only one capture bead. The capture bead may have clonally amplified DNA or other substances thereon. Alternatively, the reaction chamber may be sized and shaped to receive an approximate number of beads or solid substrates. As another example, the reaction chambers may also be filled with a porous gel or substance that is configured to control diffusion or filter fluids that may flow into the reaction chamber.

[0059] In some embodiments, sensors (*e.g.*, light detectors, photodiodes) are associated with corresponding pixel areas of a sample surface of a biosensor. As such, a pixel area is a geometrical construct that represents an area on the biosensor's sample surface for one sensor (or pixel). A sensor that is associated with a pixel area detects light emissions gathered from the associated pixel area when a desired reaction has occurred at a reaction site or a reaction chamber overlying the associated pixel area. In a flat surface embodiment, the pixel areas can overlap. In some cases, a plurality of sensors may be associated with a single reaction site or a single reaction chamber. In other cases, a single sensor may be associated with a group of reaction sites or a group of reaction chambers.

[0060] As used herein, a "biosensor" includes a structure having a plurality of reaction sites and/or reaction chambers (or wells). A biosensor may include a solid-state imaging device (*e.g.*, CCD or CMOS imager) and, optionally, a flow cell mounted thereto. The flow cell may include at least one flow channel that is in fluid communication with the reaction sites and/or the reaction chambers. As one specific example, the biosensor is configured to fluidically and electrically couple to a bioassay system. The bioassay system may deliver reactants to the reaction sites and/or the reaction chambers according to a predetermined protocol (*e.g.*, sequencing-by-synthesis) and perform a plurality of imaging events. For example, the bioassay system may direct solutions to flow along the reaction sites and/or the reaction chambers. At least one of the solutions may include four types of nucleotides having the same or different

fluorescent labels. The nucleotides may bind to corresponding oligonucleotides located at the reaction sites and/or the reaction chambers. The bioassay system may then illuminate the reaction sites and/or the reaction chambers using an excitation light source (*e.g.*, solid-state light sources, such as light-emitting diodes or LEDs). The excitation light may have a predetermined wavelength or wavelengths, including a range of wavelengths. The excited fluorescent labels provide emission signals that may be captured by the sensors.

[0061] In alternative embodiments, the biosensor may include electrodes or other types of sensors configured to detect other identifiable properties. For example, the sensors may be configured to detect a change in ion concentration. In another example, the sensors may be configured to detect the ion current flow across a membrane.

[0062] As used herein, a “cluster” is a colony of similar or identical molecules or nucleotide sequences or DNA strands. For example, a cluster can be an amplified oligonucleotide or any other group of a polynucleotide or polypeptide with a same or similar sequence. In other embodiments, a cluster can be any element or group of elements that occupy a physical area on a sample surface. In embodiments, clusters are immobilized to a reaction site and/or a reaction chamber during a base calling cycle.

[0063] As used herein, the term “immobilized,” when used with respect to a biomolecule or biological or chemical substance, includes substantially attaching the biomolecule or biological or chemical substance at a molecular level to a surface. For example, a biomolecule or biological or chemical substance may be immobilized to a surface of the substrate material using adsorption techniques including non-covalent interactions (*e.g.*, electrostatic forces, van der Waals, and dehydration of hydrophobic interfaces) and covalent binding techniques where functional groups or linkers facilitate attaching the biomolecules to the surface. Immobilizing biomolecules or biological or chemical substances to a surface of a substrate material may be based upon the properties of the substrate surface, the liquid medium carrying the biomolecule or biological or chemical substance, and the properties of the biomolecules or biological or chemical substances themselves. In some cases, a substrate surface may be functionalized (*e.g.*, chemically or physically modified) to facilitate immobilizing the biomolecules (or biological or chemical substances) to the substrate surface. The substrate surface may be first modified to have functional groups bound to the surface. The functional groups may then bind to biomolecules or biological or chemical substances to immobilize them thereon. A substance can be immobilized to a surface via a gel, for example, as described in US Patent Publ. No. US 2011/0059865 A1, which is incorporated herein by reference.

[0064] In some embodiments, nucleic acids can be attached to a surface and amplified using bridge amplification. Useful bridge amplification methods are described, for example, in U.S.

Patent No. 5,641,658; WO 2007/010251, U.S. Pat. No. 6,090,592; U.S. Patent Publ. No. 2002/0055100 A1; U.S. Patent No. 7,115,400; U.S. Patent Publ. No. 2004/0096853 A1; U.S. Patent Publ. No. 2004/0002090 A1; U.S. Patent Publ. No. 2007/0128624 A1; and U.S. Patent Publ. No. 2008/0009420 A1, each of which is incorporated herein in its entirety. Another useful method for amplifying nucleic acids on a surface is Rolling Circle Amplification (RCA), for example, using methods set forth in further detail below. In some embodiments, the nucleic acids can be attached to a surface and amplified using one or more primer pairs. For example, one of the primers can be in solution and the other primer can be immobilized on the surface (*e.g.*, 5'-attached). By way of example, a nucleic acid molecule can hybridize to one of the primers on the surface followed by extension of the immobilized primer to produce a first copy of the nucleic acid. The primer in solution then hybridizes to the first copy of the nucleic acid which can be extended using the first copy of the nucleic acid as a template. Optionally, after the first copy of the nucleic acid is produced, the original nucleic acid molecule can hybridize to a second immobilized primer on the surface and can be extended at the same time or after the primer in solution is extended. In any embodiment, repeated rounds of extension (*e.g.*, amplification) using the immobilized primer and primer in solution provide multiple copies of the nucleic acid.

[0065] In particular embodiments, the assay protocols executed by the systems and methods described herein include the use of natural nucleotides and also enzymes that are configured to interact with the natural nucleotides. Natural nucleotides include, for example, ribonucleotides (RNA) or deoxyribonucleotides (DNA). Natural nucleotides can be in the mono-, di-, or tri-phosphate form and can have a base selected from adenine (A), thymine (T), uracil (U), guanine (G) or cytosine (C). It will be understood however that non-natural nucleotides, modified nucleotides or analogs of the aforementioned nucleotides can be used. Some examples of useful non-natural nucleotides are set forth below in regard to reversible terminator-based sequencing by synthesis methods.

[0066] In embodiments that include reaction chambers, items or solid substances (including semi-solid substances) may be disposed within the reaction chambers. When disposed, the item or solid may be physically held or immobilized within the reaction chamber through an interference fit, adhesion, or entrapment. Exemplary items or solids that may be disposed within the reaction chambers include polymer beads, pellets, agarose gel, powders, quantum dots, or other solids that may be compressed and/or held within the reaction chamber. In particular embodiments, a nucleic acid superstructure, such as a DNA ball, can be disposed in or at a reaction chamber, for example, by attachment to an interior surface of the reaction chamber or by residence in a liquid within the reaction chamber. A DNA ball or other nucleic acid superstructure can be preformed and then disposed in or at the reaction chamber. Alternatively, a

DNA ball can be synthesized in situ at the reaction chamber. A DNA ball can be synthesized by rolling circle amplification to produce a concatemer of a particular nucleic acid sequence and the concatemer can be treated with conditions that form a relatively compact ball. DNA balls and methods for their synthesis are described, for example in, U.S. Patent Publication Nos.

2008/0242560 A1 or 2008/0234136 A1, each of which is incorporated herein in its entirety. A substance that is held or disposed in a reaction chamber can be in a solid, liquid, or gaseous state.

[0067] As used herein, “base calling” identifies a nucleotide base in a nucleic acid sequence. Base calling refers to the process of determining a base call (A, C, G, T) for every cluster at a specific cycle. As an example, base calling can be performed utilizing four-channel, two-channel or one-channel methods and systems described in the incorporated materials of U.S. Patent Application Publication No. 2013/0079232. In particular embodiments, a base calling cycle is referred to as a “sampling event.” In one dye and two-channel sequencing protocol, a sampling event comprises two illumination stages in time sequence, such that a pixel signal is generated at each stage. The first illumination stage induces illumination from a given cluster indicating nucleotide bases A and T in a AT pixel signal, and the second illumination stage induces illumination from a given cluster indicating nucleotide bases C and T in a CT pixel signal.

Biosensor

[0068] **Fig. 1** illustrates a cross-section of a biosensor 100 that can be used in various embodiments. Biosensor 100 has pixel areas 106', 108', 110', 112', and 114' that can each hold more than one cluster during a base calling cycle (*e.g.*, 2 clusters per pixel area). As shown, the biosensor 100 may include a flow cell 102 that is mounted onto a sampling device 104. In the illustrated embodiment, the flow cell 102 is affixed directly to the sampling device 104. However, in alternative embodiments, the flow cell 102 may be removably coupled to the sampling device 104. The sampling device 104 has a sample surface 134 that may be functionalized (*e.g.*, chemically or physically modified in a suitable manner for conducting the desired reactions). For example, the sample surface 134 may be functionalized and may include a plurality of pixel areas 106', 108', 110', 112', and 114' that can each hold more than one cluster during a base calling cycle (*e.g.*, each having a corresponding cluster pair 106A, 106B; 108A, 108B; 110A, 110B; 112A, 112B; and 114A, 114B immobilized thereto). Each pixel area is associated with a corresponding sensor (or pixel or photodiode) 106, 108, 110, 112, and 114, such that light received by the pixel area is captured by the corresponding sensor. A pixel area 106' can be also associated with a corresponding reaction site 106" on the sample surface 134 that holds a cluster pair, such that light emitted from the reaction site 106" is received by the pixel area 106' and captured by the corresponding sensor 106. As a result of this sensing

structure, in the case in which two or more clusters are present in a pixel area of a particular sensor during a base calling cycle (*e.g.*, each having a corresponding cluster pair), the pixel signal in that base calling cycle carries information based on all of the two or more clusters. As a result, signal processing as described herein is used to distinguish each cluster, where there are more clusters than pixel signals in a given sampling event of a particular base calling cycle.

[0069] In the illustrated embodiment, the flow cell 102 includes sidewalls 138, 125, and a flow cover 136 that is supported by the sidewalls 138, 125. The sidewalls 138, 125 are coupled to the sample surface 134 and extend between the flow cover 136 and the sidewalls 138, 125. In some embodiments, the sidewalls 138, 125 are formed from a curable adhesive layer that bonds the flow cover 136 to the sampling device 104.

[0070] The sidewalls 138, 125 are sized and shaped so that a flow channel 144 exists between the flow cover 136 and the sampling device 104. The flow cover 136 may include a material that is transparent to excitation light 101 propagating from an exterior of the biosensor 100 into the flow channel 144. In an example, the excitation light 101 approaches the flow cover 136 at a non-orthogonal angle.

[0071] Also shown, the flow cover 136 may include inlet and outlet ports 142, 146 that are configured to fluidically engage other ports (not shown). For example, the other ports may be from the cartridge or the workstation. The flow channel 144 is sized and shaped to direct a fluid along the sample surface 134. A height H_1 and other dimensions of the flow channel 144 may be configured to maintain a substantially even flow of a fluid along the sample surface 134. The dimensions of the flow channel 144 may also be configured to control bubble formation.

[0072] By way of example, the flow cover 136 (or the flow cell 102) may comprise a transparent material, such as glass or plastic. The flow cover 136 may constitute a substantially rectangular block having a planar exterior surface and a planar inner surface that defines the flow channel 144. The block may be mounted onto the sidewalls 138, 125. Alternatively, the flow cell 102 may be etched to define the flow cover 136 and the sidewalls 138, 125. For example, a recess may be etched into the transparent material. When the etched material is mounted to the sampling device 104, the recess may become the flow channel 144.

[0073] The sampling device 104 may be similar to, for example, an integrated circuit comprising a plurality of stacked substrate layers 120-126. The substrate layers 120-126 may include a base substrate 120, a solid-state imager 122 (*e.g.*, CMOS image sensor), a filter or light-management layer 124, and a passivation layer 126. It should be noted that the above is only illustrative and that other embodiments may include fewer or additional layers. Moreover, each of the substrate layers 120-126 may include a plurality of sub-layers. The sampling device 104 may be manufactured using processes that are similar to those used in manufacturing

integrated circuits, such as CMOS image sensors and CCDs. For example, the substrate layers 120-126 or portions thereof may be grown, deposited, etched, and the like to form the sampling device 104.

[0074] The passivation layer 126 is configured to shield the filter layer 124 from the fluidic environment of the flow channel 144. In some cases, the passivation layer 126 is also configured to provide a solid surface (*i.e.*, the sample surface 134) that permits biomolecules or other analytes-of-interest to be immobilized thereon. For example, each of the reaction sites may include a cluster of biomolecules that are immobilized to the sample surface 134. Thus, the passivation layer 126 may be formed from a material that permits the reaction sites to be immobilized thereto. The passivation layer 126 may also comprise a material that is at least transparent to a desired fluorescent light. By way of example, the passivation layer 126 may include silicon nitride (Si_2N_4) and/or silica (SiO_2). However, other suitable material(s) may be used. In the illustrated embodiment, the passivation layer 126 may be substantially planar. However, in alternative embodiments, the passivation layer 126 may include recesses, such as pits, wells, grooves, and the like. In the illustrated embodiment, the passivation layer 126 has a thickness that is about 150-200 nm and, more particularly, about 170 nm.

[0075] The filter layer 124 may include various features that affect the transmission of light. In some embodiments, the filter layer 124 can perform multiple functions. For instance, the filter layer 124 may be configured to (a) filter unwanted light signals, such as light signals from an excitation light source; (b) direct emission signals from the reaction sites toward corresponding sensors 106, 108, 110, 112, and 114 that are configured to detect the emission signals from the reaction sites; or (c) block or prevent detection of unwanted emission signals from adjacent reaction sites. As such, the filter layer 124 may also be referred to as a light-management layer. In the illustrated embodiment, the filter layer 124 has a thickness that is about 1-5 μm and, more particularly, about 2-4 μm . In alternative embodiments, the filter layer 124 may include an array of microlenses or other optical components. Each of the microlenses may be configured to direct emission signals from an associated reaction site to a sensor.

[0076] In some embodiments, the solid-state imager 122 and the base substrate 120 may be provided together as a previously constructed solid-state imaging device (*e.g.*, CMOS chip). For example, the base substrate 120 may be a wafer of silicon and the solid-state imager 122 may be mounted thereon. The solid-state imager 122 includes a layer of semiconductor material (*e.g.*, silicon) and the sensors 106, 108, 110, 112, and 114. In the illustrated embodiment, the sensors are photodiodes configured to detect light. In other embodiments, the sensors comprise light detectors. The solid-state imager 122 may be manufactured as a single chip through a CMOS-based fabrication processes.

[0077] The solid-state imager 122 may include a dense array of sensors 106, 108, 110, 112, and 114 that are configured to detect activity indicative of a desired reaction from within or along the flow channel 144. In some embodiments, each sensor has a pixel area (or detection area) that is about 1-2 square micrometer (μm^2). The array can include 500,000 sensors, 5 million sensors, 10 million sensors, or even 120 million sensors. The sensors 106, 108, 110, 112, and 114 can be configured to detect a predetermined wavelength of light that is indicative of the desired reactions.

[0078] In some embodiments, the sampling device 104 includes a microcircuit arrangement, such as the microcircuit arrangement described in U.S. Patent No. 7,595,882, which is incorporated herein by reference in the entirety. More specifically, the sampling device 104 may comprise an integrated circuit having a planar array of the sensors 106, 108, 110, 112, and 114. Circuitry formed within the sampling device 104 may be configured for at least one of signal amplification, digitization, storage, and processing. The circuitry may collect and analyze the detected fluorescent light and generate pixel signals (or detection signals) for communicating detection data to a signal processor. The circuitry may also perform additional analog and/or digital signal processing in the sampling device 104. Sampling device 104 may include conductive vias 130 that perform signal routing (*e.g.*, transmit the pixel signals to the signal processor). The pixel signals may also be transmitted through electrical contacts 132 of the sampling device 104.

[0079] The sampling device 104 is discussed in further details with respect to U.S. Nonprovisional Patent Application No. 16/874,599, titled “Systems and Devices for Characterization and Performance Analysis of Pixel-Based Sequencing,” filed May 14, 2020 (Attorney Docket No. ILLM 1011-4/IP-1750-US), which is incorporated by reference as if fully set forth herein. The sampling device 104 is not limited to the above constructions or uses as described above. In alternative embodiments, the sampling device 104 may take other forms. For example, the sampling device 104 may comprise a CCD device, such as a CCD camera, that is coupled to a flow cell or is moved to interface with a flow cell having reaction sites therein.

[0080] **Fig. 2** depicts one implementation of a flow cell 200 that contains clusters in its tiles. The flow cell 200 corresponds to the flow cell 102 of Fig. 1, *e.g.*, without the flow cover 136. Furthermore, the depiction of the flow cell 200 is symbolic in nature, and the flow cell 200 symbolically depicts various lanes and tiles therewithin, without illustrating various other components therewithin. Fig. 2 illustrates a top view of the flow cell 200.

[0081] In an embodiment, the flow cell 200 is divided or partitioned in a plurality of lanes, such as lanes 202a, 202b, ..., 202P, *i.e.*, P number of lanes. In the example of Fig. 2, the flow

cell 200 is illustrated to include 8 lanes, *i.e.*, $P = 8$ in this example, although the number of lanes within a flow cell is implementation specific.

[0082] In an embodiment, individual lanes 202 are further partitioned into non-overlapping regions called “tiles” 212. For example, Fig. 2 illustrates a magnified view of a section 208 of an example lane. The section 208 is illustrated to comprise a plurality of tiles 212.

[0083] In an example, each lane 202 comprises one or more columns of tiles. For example, in Fig. 2, each lane 202 comprises two corresponding columns of tiles 212, as illustrated within the magnified section 208. A number of tiles within each column of tiles within each lane is implementation specific, and in one example, there can be 50 tiles, 60 tiles, 100 tiles, or another appropriate number of tiles in each column of tiles within each lane.

[0084] Each tile comprises a corresponding plurality of clusters. During the sequencing procedure, the clusters and their surrounding background on the tiles are imaged. For example, Fig. 2 illustrates example clusters 216 within an example tile.

[0085] **Fig. 3** illustrates an example Illumina GA-IIx™ flow cell with eight lanes, and also illustrates a zoom-in on one tile and its clusters and their surrounding background. For example, there are a hundred tiles per lane in Illumina Genome Analyzer II and sixty-eight tiles per lane in Illumina HiSeq2000. A tile 212 holds hundreds of thousands to millions of clusters. In Fig. 3, an image generated from a tile with clusters shown as bright spots is shown at 308 (*e.g.*, 308 is a magnified image view of a tile), with an example cluster 304 labelled. A cluster 304 comprises approximately one thousand identical copies of a template molecule, though clusters vary in size and shape. The clusters are grown from the template molecule, prior to the sequencing run, by bridge amplification of the input library. The purpose of the amplification and cluster growth is to increase the intensity of the emitted signal since the imaging device cannot reliably sense a single fluorophore. However, the physical distance of the DNA fragments within a cluster 304 is small, so the imaging device perceives the cluster of fragments as a single spot 304.

[0086] The clusters and the tiles are discussed in further details with respect to U.S. Nonprovisional Patent Application No. 16/825,987, titled “TRAINING DATA GENERATION FOR ARTIFICIAL INTELLIGENCE-BASED SEQUENCING,” filed 20 March 2020 (Attorney Docket No. ILLM 1008-16/IP-1693-US);

[0087] **Fig. 4** is a simplified block diagram of the system for analysis of sensor data from a sequencing system, such as base call sensor outputs (*e.g.*, see Fig. 1). In the example of Fig. 4, the system includes a sequencing machine 400 and a configurable processor 450. The configurable processor 450 can execute a neural network-based base caller in coordination with a runtime program executed by a host processor, such as a central processing unit (CPU) 402. The sequencing machine 400 comprises base call sensors and flow cell 401 (*e.g.*, discussed with

respect to Figs. 1-3). The flow cell can comprise one or more tiles in which clusters of genetic material are exposed to a sequence of analyte flows used to cause reactions in the clusters to identify the bases in the genetic material, as discussed with respect to Figs. 1-3. The sensors sense the reactions for each cycle of the sequence in each tile of the flow cell to provide tile data. Examples of this technology are described in more detail below. Genetic sequencing is a data intensive operation, which translates base call sensor data into sequences of base calls for each cluster of genetic material sensed in during a base call operation.

[0088] The system in this example includes the CPU 402 which executes a runtime program to coordinate the base call operations, memory 403 to store sequences of arrays of tile data, base call reads produced by the base calling operation, and other information used in the base call operations. Also, in this illustration the system includes memory 404 to store a configuration file (or files), such as FPGA bit files, and model parameters for the neural network used to configure and reconfigure the configurable processor 450 and execute the neural network. The sequencing machine 400 can include a program for configuring a configurable processor and in some embodiments a reconfigurable processor to execute the neural network.

[0089] The sequencing machine 400 is coupled by a bus 405 to the configurable processor 450. The bus 405 can be implemented using a high throughput technology, such as in one example bus technology compatible with the PCIe standards (Peripheral Component Interconnect Express) currently maintained and developed by the PCI-SIG (PCI Special Interest Group). Also, in this example, a memory 460 is coupled to the configurable processor 450 by bus 461. The memory 460 can be on-board memory, disposed on a circuit board with the configurable processor 450. The memory 460 is used for high-speed access by the configurable processor 450 of working data used in the base call operation. The bus 461 can also be implemented using a high throughput technology, such as bus technology compatible with the PCIe standards.

[0090] Configurable processors, including Field Programmable Gate Arrays (FPGAs), Coarse Grained Reconfigurable Arrays (CGRAs), and other configurable and reconfigurable devices, can be configured to implement a variety of functions more efficiently or faster than might be achieved using a general-purpose processor executing a computer program. Configuration of configurable processors involves compiling a functional description to produce a configuration file, referred to sometimes as a bitstream or bit file, and distributing the configuration file to the configurable elements on the processor.

[0091] The configuration file defines the logic functions to be executed by the configurable processor, by configuring the circuit to set data flow patterns, use of distributed memory and other on-chip memory resources, lookup table contents, operations of configurable logic blocks

and configurable execution units like multiply-and-accumulate units, configurable interconnects and other elements of the configurable array. A configurable processor is reconfigurable if the configuration file may be changed in the field, by changing the loaded configuration file. For example, the configuration file may be stored in volatile SRAM elements, in non-volatile read-write memory elements, and in combinations of the same, distributed among the array of configurable elements on the configurable or reconfigurable processor. A variety of commercially available configurable processors are suitable for use in a base calling operation as described herein. Examples include commercially available products such as Xilinx Alveo™ U200, Xilinx Alveo™ U250, Xilinx Alveo™ U280, Intel/Altera Stratix™ GX2800, Intel/Altera Stratix™ GX2800, and Intel Stratix™ GX10M. In some examples, a host CPU can be implemented on the same integrated circuit as the configurable processor.

[0092] Embodiments described herein implement the multi-cycle neural network using a configurable processor 450. The configuration file for a configurable processor can be implemented by specifying the logic functions to be executed using a high-level description language (HDL) or a register transfer level (RTL) language specification. The specification can be compiled using the resources designed for the selected configurable processor to generate the configuration file. The same or similar specification can be compiled for the purposes of generating a design for an application-specific integrated circuit which may not be a configurable processor.

[0093] Alternatives for the configurable processor, in all embodiments described herein, therefore include a configured processor comprising an application specific ASIC or special purpose integrated circuit or set of integrated circuits, or a system-on-a-chip SOC device, configured to execute a neural network based base call operation as described herein.

[0094] In general, configurable processors and configured processors described herein, as configured to execute runs of a neural network, are referred to herein as neural network processors.

[0095] The configurable processor 450 is configured in this example by a configuration file loaded using a program executed by the CPU 402, or by other sources, which configures the array of configurable elements on the configurable processor 454 to execute the base call function. In this example, the configuration includes data flow logic 451 which is coupled to the buses 405 and 461 and executes functions for distributing data and control parameters among the elements used in the base call operation.

[0096] Also, the configurable processor 450 is configured with base call execution logic 452 to execute a multi-cycle neural network. The logic 452 comprises a plurality of multi-cycle execution clusters (*e.g.*, 453) which, in this example, includes multi-cycle cluster 1 through

multi-cycle cluster X. The number of multi-cycle clusters can be selected according to a trade-off involving the desired throughput of the operation, and the available resources on the configurable processor.

[0097] The multi-cycle clusters are coupled to the data flow logic 451 by data flow paths 454 implemented using configurable interconnect and memory resources on the configurable processor. Also, the multi-cycle clusters are coupled to the data flow logic 451 by control paths 455 implemented using configurable interconnect and memory resources for example on the configurable processor, which provide control signals indicating available clusters, readiness to provide input units for execution of a run of the neural network to the available clusters, readiness to provide trained parameters for the neural network, readiness to provide output patches of base call classification data, and other control data used for execution of the neural network.

[0098] The configurable processor is configured to execute runs of a multi-cycle neural network using trained parameters to produce classification data for sensing cycles of the base flow operation. A run of the neural network is executed to produce classification data for a subject sensing cycle of the base call operation. A run of the neural network operates on a sequence including a number N of arrays of tile data from respective sensing cycles of N sensing cycles, where the N sensing cycles provide sensor data for different base call operations for one base position per operation in time sequence in the examples described herein. Optionally, some of the N sensing cycles can be out of sequence if the needed according to a particular neural network model being executed. The number N can be any number greater than one. In some examples described herein, sensing cycles of the N sensing cycles represent a set of sensing cycles for at least one sensing cycle preceding the subject sensing cycle and at least one sensing cycle following the subject cycle in time sequence. Examples are described herein in which the number N is an integer equal to or greater than five.

[0099] The data flow logic 451 is configured to move tile data and at least some trained parameters of the model from the memory 460 to the configurable processor for runs of the neural network, using input units for a given run including tile data for spatially aligned patches of the N arrays. The input units can be moved by direct memory access operations in one DMA operation, or in smaller units moved during available time slots in coordination with the execution of the neural network deployed.

[0100] Tile data for a sensing cycle as described herein can comprise an array of sensor data having one or more features. For example, the sensor data can comprise two images which are analyzed to identify one of four bases at a base position in a genetic sequence of DNA, RNA, or other genetic material. The tile data can also include metadata about the images and the sensors.

For example, in embodiments of the base calling operation, the tile data can comprise information about alignment of the images with the clusters such as distance from center information indicating the distance of each pixel in the array of sensor data from the center of a cluster of genetic material on the tile.

[0101] During execution of the multi-cycle neural network as described below, tile data can also include data produced during execution of the multi-cycle neural network, referred to as intermediate data, which can be reused rather than recomputed during a run of the multi-cycle neural network. For example, during execution of the multi-cycle neural network, the data flow logic can write intermediate data to the memory 460 in place of the sensor data for a given patch of an array of tile data. Embodiments like this are described in more detail below.

[0102] As illustrated, a system is described for analysis of base call sensor output, comprising memory (*e.g.*, 460) accessible by the runtime program storing tile data including sensor data for a tile from sensing cycles of a base calling operation. Also, the system includes a neural network processor, such as configurable processor 450 having access to the memory. The neural network processor is configured to execute runs of a neural network using trained parameters to produce classification data for sensing cycles. As described herein, a run of the neural network is operating on a sequence of N arrays of tile data from respective sensing cycles of N sensing cycles, including a subject cycle, to produce the classification data for the subject cycle. The data flow logic 451 is provided to move tile data and the trained parameters from the memory to the neural network processor for runs of the neural network using input units including data for spatially aligned patches of the N arrays from respective sensing cycles of N sensing cycles.

[0103] Also, a system is described in which the neural network processor has access to the memory, and includes a plurality of execution clusters, the execution logic clusters in the plurality of execution clusters configured to execute a neural network. The data flow logic has access to the memory and to execution clusters in the plurality of execution clusters, to provide input units of tile data to available execution clusters in the plurality of execution clusters, the input units including a number N of spatially aligned patches of arrays of tile data from respective sensing cycles, including a subject sensing cycle, and to cause the execution clusters to apply the N spatially aligned patches to the neural network to produce output patches of classification data for the spatially aligned patch of the subject sensing cycle, where N is greater than 1.

[0104] **Fig. 5** is a simplified diagram showing aspects of the base calling operation, including functions of a runtime program executed by a host processor. In this diagram, the output of image sensors from a flow cell (such as those illustrated in Figs. 1-2) are provided on lines 500

to image processing threads 501, which can perform processes on images such as resampling, alignment and arrangement in an array of sensor data for the individual tiles, and can be used by processes which calculate a tile cluster mask for each tile in the flow cell, which identifies pixels in the array of sensor data that correspond to clusters of genetic material on the corresponding tile of the flow cell. To compute a cluster mask, one example algorithm is based on a process to detect clusters which are unreliable in the early sequencing cycles using a metric derived from the softmax output, and then the data from those wells/clusters is discarded, and no output data is produced for those clusters. For example, a process can identify clusters with high reliability during the first N1 (e.g., 25) base-calls, and reject the others. Rejected clusters might be polyclonal or very weak intensity or obscured by fiducials. This procedure can be performed on the host CPU. In alternative implementations, this information would potentially be used to identify the necessary clusters of interest to be passed back to the CPU, thereby limiting the storage required for intermediate data.

[0105] The outputs of the image processing threads 501 are provided on lines 502 to a dispatch logic 510 in the CPU which routes the arrays of tile data to a data cache 504 on a high-speed bus 503, or on high-speed bus 505 to the multi-cluster neural network processor hardware 520, such as the configurable processor of Fig. 4, according to the state of the base calling operation. The hardware 520 returns classification data output by the neural network to the dispatch logic 510, which passes the information to the data cache 504, or on lines 511 to threads 502 that perform base call and quality score computations using the classification data, and can arrange the data in standard formats for base call reads. The outputs of the threads 502 that perform base calling and quality score computations are provided on lines 512 to threads 503 that aggregate the base call reads, perform other operations such as data compression, and write the resulting base call outputs to specified destinations for utilization by the customers.

[0106] In some embodiments, the host can include threads (not shown) that perform final processing of the output of the hardware 520 in support of the neural network. For example, the hardware 520 can provide outputs of classification data from a final layer of the multi-cluster neural network. The host processor can execute an output activation function, such as a softmax function, over the classification data to configure the data for use by the base call and quality score threads 502. Also, the host processor can execute input operations (not shown), such as resampling, batch normalization or other adjustments of the tile data prior to input to the hardware 520.

[0107] Fig. 6 is a simplified diagram of a configuration of a configurable processor such as that of Fig. 4. In Fig. 6, the configurable processor comprises in FPGA with a plurality of high speed PCIe interfaces. The FPGA is configured with a wrapper 600 which comprises the data

flow logic described with reference to Fig. 1. The wrapper 600 manages the interface and coordination with a runtime program in the CPU across the CPU communication link 609 and manages communication with the on-board DRAM 602 (*e.g.*, memory 460) via DRAM communication link 610. The data flow logic in the wrapper 600 provides patch data retrieved by traversing the arrays of tile data on the on-board DRAM 602 for the number N cycles to a cluster 601 and retrieves process data 615 from the cluster 601 for delivery back to the on-board DRAM 602. The wrapper 600 also manages transfer of data between the on-board DRAM 602 and host memory, for both the input arrays of tile data, and for the output patches of classification data. The wrapper transfers patch data on line 613 to the allocated cluster 601. The wrapper provides trained parameters, such as weights and biases on line 612 to the cluster 601 retrieved from the on-board DRAM 602. The wrapper provides configuration and control data on line 611 to the cluster 601 provided from, or generated in response to, the runtime program on the host via the CPU communication link 609. The cluster can also provide status signals on line 616 to the wrapper 600, which are used in cooperation with control signals from the host to manage traversal of the arrays of tile data to provide spatially aligned patch data, and to execute the multi-cycle neural network over the patch data using the resources of the cluster 601.

[0108] As mentioned above, there can be multiple clusters on a single configurable processor managed by the wrapper 600 configured for executing on corresponding ones of multiple patches of the tile data. Each cluster can be configured to provide classification data for base calls in a subject sensing cycle using the tile data of multiple sensing cycles described herein.

[0109] In examples of the system, model data, including kernel data like filter weights and biases can be sent from the host CPU to the configurable processor, so that the model can be updated as a function of cycle number. A base calling operation can comprise, for a representative example, on the order of hundreds of sensing cycles. Base calling operation can include paired end reads in some embodiments. For example, the model trained parameters may be updated once every 20 cycles (or other number of cycles), or according to update patterns implemented for particular systems and neural network models. In some embodiments including paired end reads in which a sequence for a given string in a genetic cluster on a tile includes a first part extending from a first end down (or up) the string, and a second part extending from a second end up (or down) the string, the trained parameters can be updated on the transition from the first part to the second part.

[0110] In some examples, image data for multiple cycles of sensing data for a tile can be sent from the CPU to the wrapper 600. The wrapper 600 can optionally do some pre-processing and transformation of the sensing data and write the information to the on-board DRAM 602. The input tile data for each sensing cycle can include arrays of sensor data including on the order of

4000 x 3000 pixels per sensing cycle per tile or more, with two features representing colors of two images of the tile, and one or two bytes per feature per pixel. For an embodiment in which the number N is three sensing cycles to be used in each run of the multi-cycle neural network, the array of tile data for each run of the multi-cycle neural network can consume on the order of hundreds of megabytes per tile. In some embodiments of the system, the tile data also includes an array of DFC data, stored once per tile, or other type of metadata about the sensor data and the tiles.

[0111] In operation, when a multi-cycle cluster is available, the wrapper allocates a patch to the cluster. The wrapper fetches a next patch of tile data in the traversal of the tile and sends it to the allocated cluster along with appropriate control and configuration information. The cluster can be configured with enough memory on the configurable processor to hold a patch of data including patches from multiple cycles in some systems, that is being worked on in place, and a patch of data that is to be worked on when the current patch of processing is finished using a ping-pong buffer technique or raster scanning technique in various embodiments.

[0112] When an allocated cluster completes its run of the neural network for the current patch and produces an output patch, it will signal the wrapper. The wrapper will read the output patch from the allocated cluster, or alternatively the allocated cluster will push the data out to the wrapper. Then the wrapper will assemble output patches for the processed tile in the DRAM 602. When the processing of the entire tile has been completed, and the output patches of data transferred to the DRAM, the wrapper sends the processed output array for the tile back to the host/CPU in a specified format. In some embodiments, the on-board DRAM 602 is managed by memory management logic in the wrapper 600. The runtime program can control the sequencing operations to complete analysis of all the arrays of tile data for all the cycles in the run in a continuous flow to provide real time analysis.

[0113] **Fig. 7** is a diagram of a multi-cycle neural network model which can be executed using the system described herein. The example shown in Fig. 7 can be referred to as a five-cycle input, one-cycle output neural network. The inputs to the multi-cycle neural network model include five spatially aligned patches (*e.g.*, 700) from the tile data arrays of five sensing cycles of a given tile. Spatially aligned patches have the same aligned row and column dimensions (x,y) as other patches in the set, so that the information relates to the same clusters of genetic material on the tile in sequence cycles. In this example, a subject patch is a patch from the array of tile data for cycle K. The set of five spatially aligned patches includes a patch from cycle K-2 preceding the subject patch by two cycles, a patch from cycle K-1 preceding the subject patch by one cycle, a patch from cycle K+1 following the patch from the subject cycle by one cycle, and a patch from cycle K+2 following the patch from the subject cycle by two cycles.

[0114] The model includes a segregated stack 701 of layers of the neural network for each of the input patches. Thus, stack 701 receives as input, tile data for the patch from cycle K+2, and is segregated from the stacks 702, 703, 704, and 705 so they do not share input data or intermediate data. In some embodiments, all of the stacks 710-705 can have identical models, and identical trained parameters. In other embodiments, the models and trained parameters may be different in the different stacks. Stack 702 receives as input, tile data for the patch from cycle K+1. Stack 703 receives as input, tile data for the patch from cycle K. Stack 704 receives as input, tile data for the patch from cycle K-1. Stack 705 receives as input, tile data for the patch from cycle K-2. The layers of the segregated stacks each execute a convolution operation of a kernel including a plurality of filters over the input data for the layer. As in the example above, the patch 700 may include three features. The output of the layer 710 may include many more features, such as 10 to 20 features. Likewise, the outputs of each of layers 711 to 716 can include any number of features suitable for a particular implementation. The parameters of the filters are trained parameters for the neural network, such as weights and biases. The output feature set (intermediate data) from each of the stacks 701-705 is provided as input to an inverse hierarchy 720 of temporal combinatorial layers, in which the intermediate data from the multiple cycles is combined. In the example illustrated, the inverse hierarchy 720 includes a first layer including three combinatorial layers 721, 722, 723, each receiving intermediate data from three of the segregated stacks, and a final layer including one combinatorial layer 730 receiving intermediate data from the three temporal layers 721, 722, 723.

[0115] The output of the final combinatorial layer 730 is an output patch of classification data for clusters located in the corresponding patch of the tile from cycle K. The output patches can be assembled into an output array classification data for the tile for cycle K. In some embodiments, the output patch may have sizes and dimensions different from the input patches. In some embodiments, the output patch may include pixel-by-pixel data that can be filtered by the host to select cluster data.

[0116] The output classification data can then be applied to a softmax function 740 (or other output activation function) optionally executed by the host, or on the configurable processor, depending on the particular implementation. An output function different from softmax could be used (*e.g.*, making a base call output parameter according to largest output, then use a learned nonlinear mapping using context/network outputs to give base quality).

[0117] Finally, the output of the softmax function 740 can be provided as base call probabilities for cycle K (750) and stored in host memory to be used in subsequent processing. Other systems may use another function for output probability calculation, *e.g.*, another nonlinear model.

[0118] The neural network can be implemented using a configurable processor with a plurality of execution clusters so as complete evaluation of one tile cycle within the duration of the time interval, or close to the duration of the time interval, of one sensing cycle, effectively providing the output data in real time. Data flow logic can be configured to distribute input units of tile data and trained parameters to the execution clusters, and to distribute output patches for aggregation in memory.

[0119] Input units of data for a five-cycle input, one-cycle output neural network like that of Fig. 7 are described with reference to Figs. 8A and 8B for a base call operation using two-channel sensor data. For example, for a given base in a genetic sequence, the base call operation can execute two flows of analyte and two reactions that generate two channels of signals, such as images, which can be processed to identify which one of four bases is located at a current position in the genetic sequence for each cluster of genetic material. In other systems, a different number of channels of sensing data may be utilized. For example, base calling can be performed utilizing one-channel methods and systems. Incorporated materials of U.S. Patent Application Publication No. 2013/0079232 discuss base calling using various number of channels, such as one-channel, two-channels, or four-channels.

[0120] **Fig. 8A** shows arrays of tile data for five cycles for a given tile, tile M, used for the purposes of executing a five-cycle input, one-cycle output neural network. The five-cycle input tile data in this example can be written to the on-board DRAM, or other memory in the system which can be accessed by the data flow logic and, for cycle K-2, includes an array 801 for channel 1 and an array 811 for channel 2, for cycle K-1, an array 802 for channel 1 and an array 812 for channel 2, for cycle K, an array 803 for channel 1 and an array 813 for channel 2, for cycle K+1, an array 804 for channel 1 and an array 814 for channel 2, for cycle K+2, an array 805 for channel 1 and an array 815 for channel 2. Also an array 820 of metadata for the tile can be written once in the memory, in this case a DFC file, included for use as input to the neural network along with each cycle.

[0121] Although Fig. 8A discusses two-channel base calling operations, using two channels is merely an example, and base calling can be performed using any other appropriate number of channels. For example, incorporated materials of U.S. Patent Application Publication No. 2013/0079232 discuss base calling using various number of channels, such as one-channel, two-channels, or four-channels, or another appropriate number of channels.

[0122] The data flow logic composes input units, which can be understood with reference to **Fig. 8B**, of tile data that includes spatially aligned patches of the arrays of tile data for each execution cluster configured to execute a run of the neural network over an input patch. An input unit for an allocated execution cluster is composed by the data flow logic by reading spatially

aligned patches (*e.g.*, 851, 852, 861, 862, 870) from each of the arrays 801-805, 811, 815, 820 of tile data for the five input cycles, and delivering them via data paths (schematically 850) to memory on the configurable processor configured for use by the allocated execution cluster. The allocated execution cluster executes a run of the five-cycle input/one-cycle output neural network, and delivers an output patch for the subject cycle K of classification data for the same patch of the tile in the subject cycle K.

[0123] Fig. 9 is a simplified representation of a stack of a neural network usable in a system like that of Fig. 7 (*e.g.*, 701 and 720). In this example, some functions of the neural network (*e.g.*, 900, 902) are executed on the host, and other portions of the neural network (*e.g.*, 901) are executed on the configurable processor.

[0124] In an example, a first function can be batch normalization (layer 910) formed on the CPU. However, in another example, batch normalization as a function may be fused into one or more layers, and no separate batch normalization layer may be present.

[0125] A number of spatial, segregated convolution layers are executed as a first set of convolution layers of the neural network, as discussed above on the configurable processor. In this example, the first set of convolution layers applies 2D convolutions spatially.

[0126] As shown in Fig. 9, a first spatial convolution 921 is executed, followed by a second spatial convolution 922, followed by a third spatial convolution 923, and so on for a number $L/2$ of spatially segregated neural network layers in each stack (L is described with reference to Figure 7). As indicated at 923A, the number of spatial layers can be any practical number, which for context may range from a few to more than 20 in different embodiments.

[0127] For SP_CONV_0, kernel weights are stored for example in a (1,6,6,3,L) structure since there are 3 input channels to this layer. In this example, the “6” in this structure is due to storing coefficients in the transformed Winograd domain (the kernel size is 3x3 in the spatial domain but expands in the transform domain).

[0128] For other SP_CONV layers, kernel weights are stored for this example in a (1,6,6 L) structure since there are $K(=L)$ inputs and outputs for each of these layers.

[0129] The outputs of the stack of spatial layers are provided to temporal layers, including convolution layers 924, 925 executed on the FPGA. Layers 924 and 925 can be convolution layers applying 1D convolutions across cycles. As indicated at 924A, the number of temporal layers can be any practical number, which for context may range from a few to more than 20 in different embodiments.

[0130] The first temporal layer, TEMP_CONV_0 layer 824, reduces the number of cycle channels from 5 to 3, as illustrated in Fig. 7. The second temporal layer, layer 925, reduces the

number of cycle channels from 3 to 1 as illustrated in Fig. 7, and reduces the number of feature maps to four outputs for each pixel, representing confidence in each base call.

[0131] The output of the temporal layers is accumulated in output patches and delivered to the host CPU to apply for example, a softmax function 930, or other function to normalize the base call probabilities.

[0132] **Fig. 10** illustrates an alternative implementation showing a 10-input, six-output neural network which can be executed for a base calling operation. In this example, tile data for spatially aligned input patches from cycles 0 to 9 are applied to segregated stacks of spatial layers, such as stack 1001 for cycle 9. The outputs of the segregated stacks are applied to an inverse hierarchical arrangement of temporal stacks 1020, having outputs 1035(2) through 1035(7) providing base call classification data for subject cycles 2 through 7.

[0133] **Fig. 11** illustrates one implementation of the specialized architecture of the neural network-based base caller (*e.g.*, Fig. 7) that is used to segregate processing of data for different sequencing cycles. The motivation for using the specialized architecture is described first.

[0134] The neural network-based base caller processes data for a current sequencing cycle, one or more preceding sequencing cycles, and one or more successive sequencing cycles. Data for additional sequencing cycles provides sequence-specific context. The neural network-based base caller learns the sequence-specific context during training and base call them. Furthermore, data for pre and post sequencing cycles provides second order contribution of pre-phasing and phasing signals to the current sequencing cycle.

[0135] Images captured at different sequencing cycles and in different image channels are misaligned and have residual registration error with respect to each other. To account for this misalignment, the specialized architecture comprises spatial convolution layers that do not mix information between sequencing cycles and only mix information within a sequencing cycle.

[0136] Spatial convolution layers use so-called “segregated convolutions” that operationalize the segregation by independently processing data for each of a plurality of sequencing cycles through a “dedicated, non-shared” sequence of convolutions. The segregated convolutions convolve over data and resulting feature maps of only a given sequencing cycle, *i.e.*, intra-cycle, without convolving over data and resulting feature maps of any other sequencing cycle.

[0137] Consider, for example, that the input data comprises (i) current data for a current (time t) sequencing cycle to be base called, (ii) previous data for a previous (time $t-1$) sequencing cycle, and (iii) next data for a next (time $t+1$) sequencing cycle. The specialized architecture then initiates three separate data processing pipelines (or convolution pipelines), namely, a current data processing pipeline, a previous data processing pipeline, and a next data processing pipeline. The current data processing pipeline receives as input the current data for the current

(time t) sequencing cycle and independently processes it through a plurality of spatial convolution layers to produce a so-called “current spatially convolved representation” as the output of a final spatial convolution layer. The previous data processing pipeline receives as input the previous data for the previous (time $t-1$) sequencing cycle and independently processes it through the plurality of spatial convolution layers to produce a so-called “previous spatially convolved representation” as the output of the final spatial convolution layer. The next data processing pipeline receives as input the next data for the next (time $t+1$) sequencing cycle and independently processes it through the plurality of spatial convolution layers to produce a so-called “next spatially convolved representation” as the output of the final spatial convolution layer.

[0138] In some implementations, the current pipeline, one or more previous pipeline(s), and one or more next processing pipeline(s) are executed in parallel.

[0139] In some implementations, the spatial convolution layers are part of a spatial convolutional network (or subnetwork) within the specialized architecture.

[0140] The neural network-based base caller further comprises temporal convolution layers that mix information between sequencing cycles, *i.e.*, inter-cycles. The temporal convolution layers receive their inputs from the spatial convolutional network and operate on the spatially convolved representations produced by the final spatial convolution layer for the respective data processing pipelines.

[0141] The inter-cycle operability freedom of the temporal convolution layers emanates from the fact that the misalignment property, which exists in the image data fed as input to the spatial convolutional network, is purged out from the spatially convolved representations by the stack, or cascade, of segregated convolutions performed by the sequence of spatial convolution layers.

[0142] Temporal convolution layers use so-called “combinatory convolutions” that groupwise convolve over input channels in successive inputs on a sliding window basis. In one implementation, the successive inputs are successive outputs produced by a previous spatial convolution layer or a previous temporal convolution layer.

[0143] In some implementations, the temporal convolution layers are part of a temporal convolutional network (or subnetwork) within the specialized architecture. The temporal convolutional network receives its inputs from the spatial convolutional network. In one implementation, a first temporal convolution layer of the temporal convolutional network groupwise combines the spatially convolved representations between the sequencing cycles. In another implementation, subsequent temporal convolution layers of the temporal convolutional network combine successive outputs of previous temporal convolution layers.

[0144] The output of the final temporal convolution layer is fed to an output layer that produces an output. The output is used to base call one or more clusters at one or more sequencing cycles.

[0145] During a forward propagation, the specialized architecture processes information from a plurality of inputs in two stages. In the first stage, segregated convolutions are used to prevent mixing of information between the inputs. In the second stage, combinatory convolutions are used to mix information between the inputs. The results from the second stage are used to make a single inference for the plurality of inputs.

[0146] This is different from the batch mode technique where a convolution layer processes multiple inputs in a batch at the same time and makes a corresponding inference for each input in the batch. In contrast, the specialized architecture maps the plurality of inputs to the single inference. The single inference can comprise more than one prediction, such as a classification score for each of the four bases (A, C, T, and G).

[0147] In one implementation, the inputs have temporal ordering such that each input is generated at a different time step and has a plurality of input channels. For example, the plurality of inputs can include the following three inputs: a current input generated by a current sequencing cycle at time step (t), a previous input generated by a previous sequencing cycle at time step ($t-1$), and a next input generated by a next sequencing cycle at time step ($t+1$). In another implementation, each input is respectively derived from the current, previous, and next inputs by one or more previous convolution layers and includes k feature maps.

[0148] In one implementation, each input can include the following five input channels: a red image channel, a red distance channel, a green image channel, a green distance channel, and a scaling channel. In another implementation, each input can include k feature maps produced by a previous convolution layer and each feature map is treated as an input channel. In yet another example, each input can have merely one channel, two channels, or another different number of channels. Incorporated materials of U.S. Patent Application Publication No. 2013/0079232 discuss base calling using various number of channels, such as one-channel, two-channels, or four-channels.

[0149] **Fig. 12** depicts one implementation of segregated layers, each of which can include convolutions. Segregated convolutions process the plurality of inputs at once by applying a convolution filter to each input in parallel. With the segregated convolutions, the convolution filter combines input channels in a same input and does not combine input channels in different inputs. In one implementation, a same convolution filter is applied to each input in parallel. In another implementation, a different convolution filter is applied to each input in parallel. In some

implementations, each spatial convolution layer comprises a bank of k convolution filters, each of which applies to each input in parallel.

[0150] **Fig. 13A** depicts one implementation of combinatory layers, each of which can include convolutions. **Fig. 13B** depicts another implementation of the combinatory layers, each of which can include convolutions. Combinatory convolutions mix information between different inputs by grouping corresponding input channels of the different inputs and applying a convolution filter to each group. The grouping of the corresponding input channels and application of the convolution filter occurs on a sliding window basis. In this context, a window spans two or more successive input channels representing, for instance, outputs for two successive sequencing cycles. Since the window is a sliding window, most input channels are used in two or more windows.

[0151] In some implementations, the different inputs originate from an output sequence produced by a preceding spatial or temporal convolution layer. In the output sequence, the different inputs are arranged as successive outputs and therefore viewed by a next temporal convolution layer as successive inputs. Then, in the next temporal convolution layer, the combinatory convolutions apply the convolution filter to groups of corresponding input channels in the successive inputs.

[0152] In one implementation, the successive inputs have temporal ordering such that a current input is generated by a current sequencing cycle at time step (t), a previous input is generated by a previous sequencing cycle at time step ($t-1$), and a next input is generated by a next sequencing cycle at time step ($t+1$). In another implementation, each successive input is respectively derived from the current, previous, and next inputs by one or more previous convolution layers and includes k feature maps.

[0153] In one implementation, each input can include the following five input channels: a red image channel, a red distance channel, a green image channel, a green distance channel, and a scaling channel. In another implementation, each input can include k feature maps produced by a previous convolution layer and each feature map is treated as an input channel.

[0154] The depth B of the convolution filter is dependent upon the number of successive inputs whose corresponding input channels are groupwise convolved by the convolution filter on a sliding window basis. In other words, the depth B is equal to the number of successive inputs in each sliding window and the group size.

[0155] In Fig. 13A, corresponding input channels from two successive inputs are combined in each sliding window, and therefore $B = 2$. In Fig. 13B, corresponding input channels from three successive inputs are combined in each sliding window, and therefore $B = 3$.

[0156] In one implementation, the sliding windows share a same convolution filter. In another implementation, a different convolution filter is used for each sliding window. In some

implementations, each temporal convolution layer comprises a bank of k convolution filters, each of which applies to the successive inputs on a sliding window basis.

[0157] Further detail of Figs. 4-10, and variations thereof, can be found in co-pending U.S. Nonprovisional Patent Application No. 17/176,147, titled “HARDWARE EXECUTION AND ACCELERATION OF ARTIFICIAL INTELLIGENCE-BASED BASE CALLER,” filed February 15, 2021 (Attorney Docket No. ILLM 1020-2/IP-1866-US), which is incorporated by reference as if fully set forth herein.

[0158] **Fig. 14** illustrates an example tile-location based weight selection scheme used for base calling. For example, illustrated in Fig. 14 is an example flow cell 1400 that comprise a plurality of lanes 1450, each lane comprising a corresponding plurality of tiles (*e.g.*, as also discussed with respect to Figs. 1 and 2). The depiction of the flow cell 1400 is symbolic in nature, and the flow cell 1400 symbolically depicts various lanes and tiles therewithin, without illustrating various other components of the flow cell 1400. Fig. 14 illustrates a top view of the flow cell 1400 (*e.g.*, without illustrating the flow cover 136 of Fig. 1).

[0159] In an embodiment and as also discussed with respect to Fig. 2, the flow cell 1400 is divided or partitioned in a plurality of lanes, such as lanes 1450a, 1450b, 1450c, ..., 1450(P-2), 1450(P-1), and 1450P, *i.e.*, P number of lanes, where P is a positive integer. As also discussed with respect to Fig. 2, in an embodiment, individual lane 1450 is further partitioned into non-overlapping regions called tiles. In an example, each lane 1450 comprises one or more columns of tiles. For example, in Fig. 14, each lane 1450 comprises two corresponding columns of tiles, where individual tile in Fig. 14 is illustrated by a corresponding rectangular box. A number of tiles within each column of tiles within each lane is implementation specific. Each tile comprises a corresponding plurality of clusters. During the sequencing procedure, the clusters and their surrounding background on the tiles are imaged. For example, Figs. 2 and 3 illustrate examples of clusters within a tile.

[0160] In an embodiment, the tiles of the flow cell 1400 are categorized into various types, *e.g.*, based on locations of the tiles. In the example implementation of Fig. 14, individual ones of the tiles of the flow cell 1400 are categorized as edge tiles 1408, near-edge tiles 1410, or non-edge (or central) tiles 1412.

[0161] For example, tiles that are on a vertical (*e.g.*, along Y axis) and/or horizontal (*e.g.*, along X-axis) edges of the flow cell 1400 are categorized as edge tiles 1408, as illustrated in Fig. 14. Thus, an edge tile 1408 is immediate adjacent to a corresponding edge of the flow cell 1400.

[0162] Tiles that are near (*e.g.*, immediate adjacent) the edge tiles are categorized as near-edge tiles 1410. For example, near-edge tiles 1410 are one tile apart from an edge of the flow

cell 1400. Thus, an edge tile 1408 separates a corresponding near-edge tile 1410 from a corresponding edge of the flow cell 1400.

[0163] Tiles that are not edge or near-edge tiles are non-edge tiles 1412, which are also referred to as central tiles 1412. Thus, central tiles 1412 are relatively near a center of the flow cell 1400, *e.g.*, as compared to the edge tiles 1408 or near-edge tiles 1410. For example, central tiles 1414 are separated from the edges of the flow cell 1400 by edge tiles 1408 and near-edge tiles 1410.

[0164] Although the tiles of the flow cell 1400 are classified into three categories in Fig. 14 (such as edge, near-edge, and central or non-edge), such categorization is merely an example, and a different tile-location based categorization can also be used. For example, in another implementation, the tiles can be categorized as (i) edge or near edge tiles, and (ii) central tiles (*e.g.*, the edge tile and the near-edge tile categories can be merged into a single category), thereby resulting into two categorizations of the tiles.

[0165] As discussed earlier, Figs. 7 and 10 are example multi-cycle neural network models that can be used for base calling, and Fig. 9 is a simplified representation of a stack of a neural network usable in a system like that of Figs. 7 and 9. Various functions within the neural network models used for base calling use biases and weights. For example, during a convolution operation, a filter comprising one or more kernels (*e.g.*, as illustrated in Fig. 12) has corresponding plurality of weights, that are trained during a training phase of the neural network models. For example, the weights are tuned using training data generated from one or more tiles, and are used for base calling in the flow cell of Fig. 14, for example.

[0166] Base calling cycles are performed for clusters in individual tiles of the flow cell 1400. In an example, parameters relevant for base calling operations for a tile can be based on a relative location of the tile. For example, the excitation light 101 discussed with respect to Fig. 1 are directed towards the tiles of the flow cell, and different tiles can receive different amount of the excitation light 101, *e.g.*, based on a location of individual tile and/or a location of one or more light sources emitting the excitation light 101. For example, if the light source(s) emitting the excitation light 101 are vertically above the flow cell 1400, central tiles 1412 may receive a different amount of light than the edge tiles 1408 and/or the near-edge tiles 1410.

[0167] In another example, peripheral or external light (*e.g.*, ambient light from outside the biosensor 100) around the flow cell 1400 can influence the amount and/or characteristics of the excitation light 101 received by individual tiles of the flow cell 1400. Merely as an example, the edge tiles 1408 may receive the excitation light 101, along with some amount of peripheral light from outside the flow cell 1400, whereas the central tiles 1412 may mainly receive the excitation light 101.

[0168] In yet another example, individual sensors (or pixel or photodiode) included in the flow cell 1400 (*e.g.*, sensors 106, 108, 110, 112, and 114 illustrated in Fig. 1) can sense light based on a location of the corresponding sensors, which are based on a location of the corresponding tiles. For example, sensing operations performed by one or more sensors associated with the edge tiles 1408 can be influenced by peripheral light (along with the excitation light 101) relatively more than an effect of the peripheral light on sensing operations of one or more other sensors associated with the central tiles 1412.

[0169] In another example, the flow of the reactant (*e.g.*, which includes any substance that may be used to obtain a desired reaction during the base calling, such as reagents, enzymes, samples, other biomolecules, and buffer solutions) flowing to various tiles can also be influenced by a tile location. For example, tiles near a source of the reactants can receive higher amount of reactants than tiles that are further from the source.

[0170] Thus, put differently, parameters associated with base calling may be slightly different for different category of tiles. Accordingly, in an embodiment, different weight sets are used for different category of tiles, to compensate for the above discussed example tile-location dependency of the base calling process.

[0171] For example, in the implementation of Fig. 14, three candidate weight sets are used: (i) edge weight set WeT 1418 for edge tiles, (ii) near-edge weight set WnT 1420 for near-edge tiles, and (iii) central weight set WcT 1422 for central (or non-edge) edge tiles.

[0172] In an example, while training a neural network model used for base calling (such as those discussed with respect to Figs. 7, 9, and 10), the neural network model is initially trained on image data generated solely by the edge tiles 1408 (*e.g.*, training data generated from the near-edge or the central tiles are not used). The resulting weights included in the edge weight set WeT 1418.

[0173] Subsequently, the neural network model is trained on image data generated solely by near-edge tiles 1410 (*e.g.*, training data generated from the edge or the central tiles are not used), and the resulting weights are included in the near-edge weight set WnT 1420. Finally, the neural network model is trained on image data generated solely by central tiles 1412 (*e.g.*, training data generated from the edge or near-edge tiles are not used), and the resulting weights are included in the edge weight set WcT 1422.

[0174] Thus, each weight set includes a corresponding plurality of weights for configuring the neural network model, where the configured neural network is to process sensor data from the corresponding category of tiles. For example, as discussed with respect to Figs. 7, 9, 10, and 11, the topology of the neural network model includes (i) one or more spatial layers that do not combine sensor data and resulting feature maps between the successive sensing cycles, and (ii)

temporal layers that combine resulting feature maps between the successive sensing cycles. Thus, each weight set includes corresponding spatial weights for the spatial layers and corresponding temporal weights for the temporal layers. For example, the edge weight set WeT 1418 for the edge tiles includes corresponding first one or more spatial weights for the spatial layers and corresponding first one or more temporal weights for the temporal layers. Similarly, the central weight set WeT 1422 for the central tiles includes corresponding second one or more spatial weights for the spatial layers and corresponding second one or more temporal weights for the temporal layers.

[0175] During the inference phase when the base calling cycles are performed, when bases within clusters of the edge tiles are to be called, the neural network model is configured with the edge weight set WeT 1418, and sensor data from the edge tiles are used for base calling operation. Similarly, when bases within clusters of the near-edge tiles are to be called, the neural network model is configured with the near-edge weight set WnT 1420, and sensor data from the near-edge tiles are used for the base calling operation. Finally, when bases within clusters of the central tiles are to be called, the neural network model is configured with the central weight set WeT 1422, and sensor data from the central tiles are used for the base calling operation.

[0176] **Fig. 15** illustrates another example tile-location based weight selection scheme used for base calling. For example, illustrated in Fig. 15 is the flow cell 1400 that comprise the plurality of lanes 1450a, 1450b, 1450c, ..., 1450(P-2), 1450(P-1), and 1450P, where each lane includes a corresponding plurality of tiles.

[0177] In the example of Fig. 15, each tile of the flow cell 1400 is categorized based on a location of a corresponding lane in which the tile belongs. For example, top one or more lanes of the flow cell 1400 (such as lanes 1450P and 1450(P-1)) are categorized as top peripheral lanes, bottom one or more lanes of the flow cell 1400 (such as lanes 1450a and 1450b) are categorized as bottom peripheral lanes, and central one or more lanes of the flow cell 1400 (such as lanes 1450c and 1450(P-2)) are categorized as central lanes. Note that a number of lanes belonging to each category is merely an example, and variations may be possible. For example, instead of two lanes, each peripheral lane category can include one corresponding lane, or three corresponding lane, or the like.

[0178] Tiles within the top peripheral lanes are categorized as top peripheral lane tiles 1508a, tiles within the bottom peripheral lanes are categorized as bottom peripheral lane tiles 1508b, and tiles within the central lanes are categorized as central lane tiles 1510.

[0179] For reasons discussed with respect to Fig. 14, in an embodiment, tiles within various categories of lanes in the flow cell of Fig. 15 can be assigned different weight sets. For example, in the implementation of Fig. 15, two candidate weight sets are used: (i) peripheral weight set

WpL 1504 for peripheral lane tiles 1508a, 1508b (*e.g.*, tiles belonging to the top and the bottom peripheral lanes), and (ii) central weight set WcL 1506 for central lane tiles 1510.

[0180] For example, while training a neural network model used for base calling (such as those discussed with respect to Figs. 7, 9, and 10), the neural network model is initially trained on image data generated solely by peripheral lane tiles 1508a, 1508b (*e.g.*, training data generated from the central lane tiles 1510 are not used). The resulting weights are included in the peripheral weight set WpL 1504.

[0181] Subsequently, the neural network model is trained on image data generated solely by central lane tiles 1510 (*e.g.*, training data generated from the peripheral lane tiles 1508a, 1508b are not used), and the resulting weights are included in the central weight set WcL 1506.

[0182] During the inference phase when the base calling cycles are performed, when bases within clusters of the peripheral lane tiles 1508 are to be called, the neural network model is configured with weights from the peripheral weight set WpL 1504, and sensor data from the peripheral lane tiles 1508 are used for the base calling operation. Similarly, when bases within clusters of the central lane tiles 1510 are to be called, the neural network model is configured with the weights from the central weight set WcL 1506, and sensor data from the central lane tiles 1510 are used for the base calling operation.

[0183] **Fig. 16** illustrates yet another example tile-location based weight selection scheme used for base calling. For example, illustrated in Fig. 16 is the flow cell 1400 that comprise the plurality of lanes 1450a, 1450b, 1450c, ..., 1450(P-2), 1450(P-1), and 1450P, where each lane includes a corresponding plurality of tiles.

[0184] In the example of Fig. 16, the flow cell 1400 is divided into a plurality of segments or sections, based on the imaginary dotted lines 1603 (*i.e.*, the dotted lines 1603 are for categorization, and are not actually present in the flow cell). For example, the flow cell 1400 is divided in a top-left section 1610TL (weight set WTL), a top-central section 1610TC (weight set WTC), a top-right section 1610TR (weight set WTR), a middle-left section 1610ML (weight set WML), a central section 1610C (weight set WC), a middle-right section 1610MR (weight set WMR), a bottom-left section 1610BL (weight set WML), a bottom-central section 1610BC (weight set WBC), and a bottom-right section 1610BR (weight set WBR). Each tile of the flow cell 1400 is categorized based on a section in which the tile belongs.

[0185] For reasons similar to those discussed with respect to Fig. 14, in an embodiment, tiles within various sections of Fig. 16 are assigned corresponding weight sets. For example, in the implementation of Fig. 16, tiles in the top-left section 1610TL are assigned top-left weight set WTL, tiles in the top-central section 1610TC are assigned top-central weight set WTC, tiles in the top-right section 1610TR are assigned top-right weight set WTR, tiles in the middle-left

section 1610ML are assigned middle-left weight set WML, tiles in the central section 1610C are assigned central weight set WC, tiles in the middle-right section 1610MR are assigned middle-right weight set WMR, tiles in the bottom-left section 1610BL are assigned bottom-left weight set WML, tiles in the bottom-central section 1610BC are assigned bottom-central weight set WBC, and tiles in the bottom-left section 1610BL are assigned bottom-left weight set WBL.

[0186] For example, while training a neural network model used for base calling (such as those discussed with respect to Figs. 7, 9, and 10), the neural network model is initially trained on sensor data generated solely by the tiles on the top-left section 1610TL (*e.g.*, sensor data from other categories of tiles are not used), and the resulting weights are included in the top-left weight set WTL. This process is repeated for tiles of various other sections, to generate the various candidate weight sets, such as the top-central weight set WTC, the top-right weight set WTR, the middle-left weight set WML, the central weight set WC, the middle-right weight set WMR, the bottom-left weight set WML, the bottom-central weight set WBC, and the bottom-left section weight set WBL.

[0187] During the inference phase when the base calling cycles are performed, when bases within clusters of the tiles within the top-left section 1610TL are to be called, the neural network model is configured with weights within the corresponding top-left weight set WTL, and sensor data from the tiles of the top-left section 1610TL are used for the base calling operation. This process is similarly repeated for tiles of various other sections.

[0188] In Fig. 16, the flow cell 1400 is sectioned in 9 different sections. However, the flow cell 1400 can be partitioned in a different number of sections, such as four sections that include a top-left quadrant, top-right quadrant, bottom-left quadrant, and a bottom-right quadrant, for example.

[0189] **Fig. 17A** illustrates an example of fading, in which signal intensity is decreased as a function of cycle number is a sequencing run of a base calling operation. Fading is an exponential decay in fluorescent signal intensity as a function of cycle number. As the sequencing run progress, the analyte strands are washed excessively, exposed to laser emissions that create reactive species, and subject to harsh environmental conditions. All of these lead to a gradual loss of fragments in each analyte, decreasing its fluorescent signal intensity. Fading is also called dimming or signal decay. Fig. 17A illustrates one example of fading 1700. In Fig. 17A, the intensity values of analyte fragments with AC microsatellites show exponential decay.

[0190] **Fig. 17B** conceptually illustrates a decreasing signal-to-noise ratio as cycles of sequencing progress. For example, as sequencing proceeds, accurate base calling becomes increasingly difficult, because signal strength decreases and noise increases, resulting in a substantially decreased signal-to-noise ratio. Physically, it was observed that later synthesis steps

attach tags in a different position relative to the sensor than earlier synthesis steps. When the sensor is below a sequence that is being synthesized, signal decay results from attaching tags to strands further away from the sensor in later sequencing steps than in earlier steps. This causes signal decay with progression of sequencing cycles. In some designs, where the sensor is above the substrate that holds cluster, signal could increase, instead of decay, as sequencing proceeds.

[0191] In the flow cell design investigated, while the signal decays, noise grows. Physically, phasing and pre-phasing increase noise as sequencing proceeds. Phasing refers to steps in sequencing in which tags fail to advance along the sequence. Pre-phasing refers to sequencing steps in which tags jump two positions forward instead of one, during a sequencing cycle. Phasing and pre-phasing are both relatively infrequent, on the order of once in 500 to 1000 cycles. Phasing is slightly more frequent than pre-phasing. Phasing and pre-phasing impact individual strands in a cluster that is producing intensity data, so the intensity noise distribution from a cluster accumulates in a binomial, trinomial, quadrinomial, etc. expansion as sequencing proceeds.

[0192] Further detail of fading, signal decay, and decrease in signal-to-noise ratio, and Figs. 17A and 17B can be found in U.S. Nonprovisional Patent Application No. 16/874,599, titled "Systems and Devices for Characterization and Performance Analysis of Pixel-Based Sequencing," filed May 14, 2020 (Attorney Docket No. ILLM 1011-4/IP-1750-US), which is incorporated by reference as if fully set forth herein.

[0193] Thus, during base calling, a reliability or quality of the base calling (*e.g.*, a probability of the called base being correct) can be based on a base calling cycle number for which the current base is being called. Thus, in addition to, or instead of, being dependent on location of tiles (*e.g.*, as discussed with respect to Figs. 14, 15, 16), weight sets can also be based on a current cycle number for which a base calling operation is being performed. **Fig. 18** illustrates an example base calling cycle-number based weight selection scheme used for base calling.

[0194] For example, Fig. 18 is directed towards base calling run for an example tile M. Assume that there are N number of base calling cycles, during which strands in various clusters in the example tile M are to be identified. As discussed, due to the factors discussed with respect to Figs. 17A and 17B and/or various other factors, signal intensity detected by the biosensors (*e.g.*, sensors 106, 108, 110, 112, and 114 of Fig. 1) changes (*e.g.*, decays) as a function of base calling cycle number. For example, assume that the N base calling sensing cycles are divided into three subseries of cycles, such as (a) initial sensing cycles 1 to N₁, (b) intermediate sensing cycles (N₁+1) to N₂, and (c) final sensing cycles (N₂+1) to N, as illustrated in Fig. 18, where N > N₂ > N₁, and N, N₁, N₂ are positive integers. Thus, the N number of sensing cycles are

divided in the three subseries of cycles, although the N number of sensing cycles can be divided into a different number (such as 2, 4, or higher) number of three subseries of cycles as well.

[0195] Note that a number of sensing cycles in each of the above discussed three subseries of cycles may, or may not, be equal, and is implementation specific. Merely as an example and without limiting the scope of this disclosure, if N is 100, then the 100 cycles can be divided into subseries comprising 30 initial cycles, 30 intermediate cycles, and 40 final cycles. That is, in this simple example, $N1 = 30$ and $N2 = 60$.

[0196] As discussed with respect to Figs. 17a and 17B, an average level of the signal intensity received by the base caller from the biosensors in cycle number N1, for example, can be different from an average level of the signal intensity received by the base caller from the biosensors in cycle number N. Thus, a neural network model trained for cycle number N1, for example, may not provide satisfactory results for cycle number N.

[0197] Accordingly, the neural network model used for base calling (such as those discussed with respect to Figs. 7, 9, and 10) can be trained for specific subseries of cycles. For example, the neural network model is initially trained on sensor data generated solely during sensing cycles 1 to N1, and the resulting weights are included in the first cycle subseries weight set W(1-N1) 1810a. Subsequently, the neural network model is trained on sensor data generated solely during sensing cycles (N1+1) to N2, and the resulting weights are included in the second cycle subseries weight set W(N1-N2) 1810b. Finally, the neural network model is trained on sensor data generated solely during sensing cycles (N2+1) to N, and the resulting weights are included in third cycle subseries weight set W(N2-N) 1810c. Note that, for example, in the first cycle subseries weight set W(1-N1) 1810a, the phrase (1-N1) is a cycle index, which implies that this weight set is relevant to sensing cycles 1 to N1. It may be noted that in the example of Fig. 18, the base calling operation is performed using sensor data from one or more channels (such as one channel, two channels, three channels, four channels, or a higher number of channels), and for a given cycle, the weights may be applicable for sensor data from all such channel(s).

[0198] During the inference phase, when bases are to be called for cycles 1 to N1, the neural network model is configured with the first cycle subseries weight set W(1-N1) 1810a. Similarly, when bases are to be called for cycles (N1+1) to N2, the neural network model is configured with the second cycle subseries weight set W(N1-N2) 1810b. Finally, when bases are to be called for cycles N2 to N3, the neural network model is configured with the third cycle subseries weight set W(N2-N3) 1810c.

[0199] Figs. 14, 15, 16 illustrate various examples of weight set selection based on location of the tiles. Thus, these figures illustrate various examples of weight set selection based on spatial progression of the base calling operation through locations of the tiles on the biosensor.

On the other hand, Fig. 18 illustrate an example of weight set selection based on temporal progression of the base calling operation through subseries of sensing cycles in a series of sensing cycles 1 to N. Fig. 19 combines concept of weight set selection based on spatial tile locations (*e.g.*, as discussed with respect to Figs. 14-16) with the concept of weight set selection based on temporal progression of the base calling cycles (*e.g.*, as discussed with respect to Fig. 18). Thus, **Fig. 19** illustrates an example weight selection scheme based on (i) temporal progression of the base calling cycle numbers and (ii) spatial locations of the tiles.

[0200] For example, Fig. 19 illustrates a first tile M1 and a second tile M2. Assume that tile M1 is a first category of tile, and tile M2 is a second category of tile. Merely as an example, tile M1 can be an edge tile 1408 of Fig. 14, and tile M2 can be a central tile 1412 of Fig. 14. Thus, weight set(s) used for base calling strands within clusters in the tile M1 would be different from weight set(s) used for base calling strands within clusters in the tile M2, *e.g.*, as discussed with respect to Figs. 14, 15, and 16.

[0201] Similar to Fig. 18, in Fig. 19 assume that there are N number of base calling cycles, during which strands in various clusters in the tiles M1 and M2 are to be identified. Furthermore, similar to Fig. 18, in Fig. 19 assume that the N base calling sensing cycles are divided into three subseries of cycles, such as (a) initial sensing cycles 1 to N1, (b) intermediate sensing cycles (N1+1) to N2, and (c) final sensing cycles (N2+1) to N, where $N > N2 > N1$, and N, N1, N2 are positive integers, although the N number of sensing cycles can be divided into a different number (such as 2, 4, or higher) number of subseries of cycles as well in other examples.

[0202] In an example, the neural network model used for base calling (such as those discussed with respect to Figs. 7, 9, and 10) can be trained for specific subseries of cycles and for specific tiles. For example, the neural network model is initially trained on sensor data generated solely during sensing cycles 1 to N1 and solely for edge tiles 1408, and the resulting weight set is labelled as “Weight set (eT,(1-N1)).” Note that the phrase “eT” in this weight set is a tile category or tile location index, which implies that this weight set is specifically for edge tiles 1408. Also, the phrase “(1-N1)” in this weight set is a cycle index, which implies that this weight set is specifically for sensing cycles 1 to N1.

[0203] Similarly, the neural network model is then trained on sensor data generated solely during sensing cycles (N1+1) to N2 and solely for edge tiles 1408, and the resulting weight set is labelled as “Weight set (eT,(N1-N2)).” Here too the phrase “eT” is the tile location or tile category index, which implies that this weight set is specifically for edge tiles 1408. Similarly, the phrase “(N1-N2)” in this weight set is the cycle index, which implies that this weight set is specifically for sensing cycles (N1+1) to N2.

[0204] Similarly, the neural network model is then trained on sensor data generated solely during sensing cycles $(N2+1)$ to N and solely for edge tiles 1408, and the resulting weight set is labelled as “Weight set ($eT, (N2-N)$).” Here too the phrase “ eT ” is the tile location index, which implies that this weight set is specifically for edge tiles 1408. Similarly, the phrase “ $(N2-N)$ ” in this weight set is the cycle index, which implies that this weight set is specifically for sensing cycles $(N2+1)$ to N .

[0205] Furthermore, the neural network model is trained on sensor data generated solely during sensing cycles 1 to $N1$ and solely for central tiles 1412, and the resulting weight set is labelled as “Weight set ($cT, (1-N1)$).” Note that the phrase “ cT ” in this weight set is the tile location index, which implies that this weight set is specifically for central tiles 1412. Also, the phrase “ $(1-N1)$ ” in this weight set is a cycle index, which implies that this weight set is specifically for sensing cycles 1 to $N1$.

[0206] Similarly, the neural network model is then trained on sensor data generated solely during sensing cycles $(N1+1)$ to $N2$ and solely for central tiles 1412, and the resulting weight set is labelled as “Weight set ($cT, (N1-N2)$).” Here too the phrase “ cT ” is the tile location index, which implies that this weight set is specifically for central tiles 1412. Similarly, the phrase “ $(N1-N2)$ ” in this weight set is the cycle index, which implies that this weight set is specifically for sensing cycles $(N1+1)$ to $N2$.

[0207] Similarly, the neural network model is then trained on sensor data generated solely during sensing cycles $(N2+1)$ to N and solely for central tiles 1412, and the resulting weight set is labelled as “Weight set ($cT, (N2-N)$).” Here too the phrase “ cT ” is the tile location index, which implies that this weight set is specifically for central tiles 1412. Similarly, the phrase “ $(N2-N)$ ” in this weight set is the cycle index, which implies that this weight set is specifically for sensing cycles $(N2+1)$ to N .

[0208] During the inference phase, when bases are to be called for cycles 1 to $N1$ and for tile $M1$ (*e.g.*, which is an edge tile 1408 in the example of Fig. 19), the neural network model is configured with the weight set ($eT, (1-N1)$). Similarly, when bases are to be called for cycles $(N1+1)$ to $N2$ and for tile $M1$, the neural network model is configured with the weight set ($eT, (N1-N2)$). Also, when bases are to be called for cycles $(N2+1)$ to N and for tile $M1$, the neural network model is configured with the weight set ($eT, (N2-N)$).

[0209] Similarly, when bases are to be called for cycles 1 to $N1$ and for tile $M2$ (*e.g.*, which is a central tile 1412 in the example of Fig. 19), the neural network model is configured with the weight set ($cT, (1-N1)$). Similarly, when bases are to be called for cycles $(N1+1)$ to $N2$ and for tile $M2$, the neural network model is configured with the weight set ($cT, (N1-N2)$). Also, when

bases are to be called for cycles $(N2+1)$ to N and for tile $M2$, the neural network model is configured with the weight set $(cT, (N2-N))$.

[0210] Fig. 20 illustrates another example weight selection scheme based on (i) temporal progression of the base calling cycle numbers and (ii) spatial locations of the tiles. The tile categorization illustrated in Fig. 20 is similar to that illustrated in Fig. 14. For example, referring to Figs. 14 and 20, the edge tiles 1408 are illustrated with diagonal lines therewithin, the near-edge tiles 1410 are illustrated with cross-hatching therewithin, and the central tiles 1412 are illustrated with dots or grey shaded therewithin.

[0211] Three boxes 1908, 1910, and 1912 are also illustrated in Fig. 20. Referring to the box 1908, illustrated are weight sets that are specific for edge tiles 1408 and for various subseries of sensing cycles. For example, weight set $(eT, (1-N1))$ is specifically for edge tiles 1408 and sensing cycles 1 to $N1$. Weight set $(eT, (N1-N2))$ is specifically for edge tiles 1408 and sensing cycles $(N1+1)$ to $N2$. Weight set $(eT, (N2-N))$ is specifically for edge tiles 1408 and sensing cycles $(N2+1)$ to N .

[0212] Similarly, referring to the box 1910, illustrated are weight sets that are specific for near edge tiles 1410 and for various subseries of sensing cycles. For example, weight set $(nT, (1-N1))$ is specifically for near edge tiles 1410 and sensing cycles 1 to $N1$. Weight set $(nT, (N1-N2))$ is specifically for near edge tiles 1410 and sensing cycles $(N1+1)$ to $N2$. Weight set $(nT, (N2-N))$ is specifically for near edge tiles 1410 and sensing cycles $(N2+1)$ to N .

[0213] Similarly, referring to box 1912, illustrated are weight sets that are specific for central tiles 1412 and for various subseries of sensing cycles. For example, weight set $(cT, (1-N1))$ is specifically for central tiles 1412 and sensing cycles 1 to $N1$. Weight set $(cT, (N1-N2))$ is specifically for central tiles 1412 and sensing cycles $(N1+1)$ to $N2$. Weight set $(cT, (N2-N))$ is specifically for central tiles 1412 and sensing cycles $(N2+1)$ to N .

[0214] Fig. 21A illustrates another example weight selection scheme based on (i) temporal progression of the base calling cycle numbers and (ii) spatial locations of the tiles. The tile categorization illustrated in Fig. 21A is similar to that illustrated in Fig. 15. For example, referring to Figs. 15 and 21, the peripheral lane tiles 1508 (which are a combination of top peripheral lane tiles 1508a and bottom peripheral lane tiles 1508b of Fig. 15) are illustrated with diagonal lines therewithin, and the central lane tiles 1510 are illustrated with dotted or grey shaded boxes.

[0215] Two boxes 2110 and 2112 are also illustrated in Fig. 21A. Referring to box 2110, illustrated are weight sets that are specific for peripheral lane tiles 1508 and for various subseries of sensing cycles. For example, weight set $(pl, (1-N1))$ is specifically for peripheral lane tiles 1508 and sensing cycles 1 to $N1$. Weight set $(pl, (N1-N2))$ is specifically for peripheral lane tiles

1508 and sensing cycles (N1+1) to N2. Weight set (pl,(N2-N)) is specifically for peripheral lane tiles 1508 and sensing cycles (N2+1) to N.

[0216] Similarly, referring to box 2112, illustrated are weight sets that are specific for central lane tiles 1510 and for various subseries of sensing cycles. For example, weight set (cl,(1-N1)) is specifically for central lane tiles 1510 and sensing cycles 1 to N1. Weight set (cl,(N1-N2)) is specifically for central lane tiles 1510 and sensing cycles (N1+1) to N2. Weight set (cl,(N2-N)) is specifically for central lane tiles 1510 and sensing cycles (N2+1) to N.

[0217] In an embodiment and as discussed above, each of the weight set (pl,(1-N1)), weight set (pl,(N1-N2)), weight set (pl,(N2-N)), weight set (cl,(1-N1)), weight set (cl,(N1-N2)), weight set (cl,(N2-N)) comprises corresponding weights. For example, the weight set (pl,(1-N1)) comprises a first plurality of weights for configuring a corresponding plurality of spatial and temporal layers (*e.g.*, see Figs. 7 and 9 for examples of such layers), the weight set (pl,(N1-N2)) comprises a second plurality of weights for configuring the corresponding plurality of spatial and temporal layers, the weight set (pl,(N2-N)) comprises a third plurality of weights for configuring the corresponding plurality of spatial and temporal layers, the weight set (cl,(1-N1)) comprises a fourth plurality of weights for configuring the corresponding plurality of spatial and temporal layers, the weight set (cl,(N1-N2)) comprises a fifth plurality of weights for configuring the corresponding plurality of spatial and temporal layers, and the weight set (cl,(N2-N)) comprises a sixth plurality of weights for configuring the corresponding plurality of spatial and temporal layers.

[0218] At least one weight in the first plurality of weights is different from a corresponding weight in the second plurality of weights (in some examples, two weight sets can have one or more common or same weights). At least one weight in the second plurality of weights is different from a corresponding weight in the third plurality of weights, at least one weight in the third plurality of weights is different from a corresponding weight in the fourth plurality of weights, and so on. In an embodiment, one or more weights in the various weight sets are quantized using different scaling factors.

[0219] Because various weight sets are associated with corresponding sequencing cycles, in an example, weights in various weight sets respectively correspond to various sequencing chemistries, sequencing configurations, and/or sequencing assays. For example, the weight set (pl,(1-N1)), weight set (pl,(N1-N2)), and weight set (pl,(N2-N)) respectively correspond to first, second, and third sequencing chemistries (*e.g.*, which are respectively used during sequencing cycles 1 to N1, (N1+1) to N2, and (N2+1) to N). The weight set (pl,(1-N1)), weight set (pl,(N1-N2)), and weight set (pl,(N2-N)) respectively correspond to first, second, and third sequencing

assays. The weight set (pl,(1-N1)), weight set (pl,(N1-N2)), and weight set (pl,(N2-N)) respectively correspond to first, second, and third sequencing configurations.

[0220] Fig. 21B illustrates yet another example weight selection scheme based on (i) temporal progression of the base calling cycle numbers and (ii) spatial locations of the tiles. The tile categorization illustrated in Fig. 21B is similar to that illustrated in Fig. 16. For example, referring to Figs. 16 and 21B, the flow cell 1400 is divided in a top-left section 1610TL, a top-central section 1610TC, a top-right section 1610TR, a middle-left section 1610ML, a central section 1610C, a middle-right section 1610MR, a bottom-left section 1610BL, a bottom-central section 1610BC, and a bottom-right section 1610BR. Each tile of the flow cell 1400 is categorized based on a section in which the tile belongs.

[0221] Fig. 21B also illustrates a table 2150 that includes various weights for tiles of various sections, as well as for various subseries of sensing cycles 1 to N. For example, referring to the first row of table 2150, weight set (TL,(1-N1)) is specifically for the tiles of the top-left section 1610TL and sensing cycles 1 to N1. Weight set (TL,(N1-N2)) is specifically for the tiles of the top-left section 1610TL and sensing cycles (N1+1) to N2. Weight set (TL,(N2-N)) is specifically for the tiles of the top-left section 1610TL and sensing cycles (N2+1) to N.

[0222] Similarly, referring to the second row of table 2150, weight set (TC,(1-N1)) is specifically for the tiles of the top-central section 1610TC and sensing cycles 1 to N1. Weight set (TC,(N1-N2)) is specifically for the tiles of the top-central section 1610TC and sensing cycles (N1+1) to N2. Weight set (TC,(N2-N)) is specifically for the tiles of the top-central section 1610TC and sensing cycles (N2+1) to N. Similarly, various other rows of the table 2150 include weight sets for tiles of various other sections and for various subseries of sensing cycles, and will be apparent to those skilled in the art based on the above discussion.

[0223] Fig. 22 illustrates one implementation of base calling operations 2200, in which weight sets used for the base calling are selected based on spatial tile information and temporal subseries of sensing cycle information.

[0224] For the base calling operations 2200 of Fig. 22, the tiles of the flow cell 1400 are assumed to be categorized in accordance with the examples of Figs. 15 and 21A. Such a tile categorization is not intended to limit the scope of this disclosure, and the base calling operations 2200 can also be applied to any other type of tile categorization, such as any of those discussed with respect to Figs. 14, 16, 20, 21B, and/or any other tile categorization envisioned by those skilled in the art based on the teachings of this disclosure.

[0225] Furthermore, for the base calling operations 2200 of Fig. 22, the N number of sensing cycles are assumed to be divided into three subseries of cycles comprising (a) cycles 1 to N1, (b) cycles (N1+1) to N2, and (c) cycles (N2+1) to N, as discussed with respect to Figs. 18-21B.

Again, such a division of the sensing cycles is not intended to limit the scope of this disclosure, and the base calling operations 2200 can also be applied to any other type of subdivision of sensing cycles that can be envisioned by those skilled in the art based on the teachings of this disclosure.

[0226] In Fig. 22, base calling operations 1a-6a are specifically for peripheral lane tiles and cycles 1 to N1. Similarly, base calling operations 1b-6b are specifically for central lane tiles and cycles 1 to N1. The operations of 1a-6a and for 1b-6b can be repeated for cycles (N1+1) to N2, and can be further repeated for cycles (N2+1) to N, although such repetitions are not illustrated in detail Fig. 22. Such repetitions for cycles (N1+1) to N2, and further for cycles (N2+1) to N will be appreciated by those skilled in the art, based on the discussion of the operations 1a-6a and 1b-6b for cycles 1 to N1.

[0227] At action 1a, the data flow logic 451 (*e.g.*, see Fig. 4) receives cluster sensor data and weight set (pl,(1-N1)) (see Fig. 21A) for peripheral lane tiles 1508 and for cycles 1 to N1. The cluster data includes sequencing images that depict intensity emissions of clusters within the peripheral lane tiles 1508 at sequencing cycles 1 to N1 of the sequencing run, as discussed above. At action 2a, the data flow logic 451 forwards the cluster data and the weight set (pl,(1-N1)) for the peripheral lane tiles 1508 and for cycles 1 to N1 to a neural network based base-caller 2308 (*e.g.*, example of which are illustrated in Figs. 7, 9, 10) executed by the configurable processor 450 (*e.g.*, see Fig. 4). The cluster data and the weight set (pl,(1-N1)) for the peripheral lane tiles 1508 and for cycles 1 to N1 are loaded in the neural network based base-caller 2308. Also, although not illustrated in Fig. 22, a topology of the neural network model is also loaded from the memory, via the data flow logic 451, to the configurable processor 450.

[0228] At action 3a, the configurable processor 450 configures the topology of the neural network running on the configurable processor 450 with the loaded weight set (pl,(1-N1)). The neural network-based base caller 2308, configured with the loaded weight set (pl,(1-N1)), generates representations (*e.g.*, feature maps) from the cluster data based on the loaded weight set (pl,(1-N1)) (*e.g.*, by processing the cluster data through its configured spatial and temporal convolution layers), and produces base call classification data (*e.g.*, base call classification scores) for the plurality of clusters within the peripheral lane tiles 1508 and for the sequencing cycles 1 to N1 based on the representations. For example, the neural network-based base caller 2308 applies the loaded weight set (pl,(1-N1)) on the cluster data, to generate the base call classification data. In one implementation, the base call classification scores are unnormalized, for example, they are not subjected to exponential normalization by a softmax function.

[0229] At action 4a, the configurable processor 450 sends to the data flow logic 451 the base call classification data for the clusters within the peripheral lane tiles 1508 and for cycles 1 to

N1. At action 5a, the data flow logic 451 provides, to the host processor 2304, the base call classification scores for the clusters within the peripheral lane tiles 1508 and for cycles 1 to N1.

[0230] At action 6a, the host processor 2304 normalizes the unnormalized base call classification scores (*e.g.*, by applying a softmax function, block 740 of Fig. 7 or 930 of Fig. 9), and generates normalized base call classification scores, *i.e.*, base calls, for strands within the clusters of the peripheral lane tiles 1508 and for cycles 1 to N1.

[0231] Thus, in operations 1a-6a, the system base calls strands within the clusters of the peripheral lane tiles 1508 and for cycles 1 to N1, using the weight set (pl,(1-N1)) that is specifically trained for the peripheral lane tiles 1508 and for cycles 1 to N1. Note that the operations 1a-6a depict a high level and simplified version of base calling operations, and may not illustrate one or more other operations that may be performed for base calling. Further detail of base calling operations may be found in U.S. Provisional Patent Application No. 63/072,032, titled "DETECTING AND FILTERING CLUSTERS BASED ON ARTIFICIAL INTELLIGENCE-PREDICTED BASE CALLS," filed 28 August 2020 (Attorney Docket No. ILLM 1018-1/IP-1860-PRV), which is incorporated by reference as if fully set forth herein.

[0232] Operations 1a-6a are specifically for base calling for strands within clusters of the peripheral lane tiles 1508 and for cycles 1 to N1. These operations are repeated as operations 1b-6b, but for clusters within the central lane tiles 1510 and for cycles 1 to N1. For example, at action 1b, the data flow logic 451 receives cluster data and weight set (cl,(1-N1)) (see Fig. 21A) for the central lane tiles 1510 and for cycles 1 to N1. The cluster data includes sequencing images that depict intensity emissions of clusters within the central lane tiles 1510 at sequencing cycles 1 to N1 of the sequencing run, as discussed above. At action 2b, the data flow logic 451 forwards the cluster data and the weight set (cl,(1-N1)) for the central lane tiles 1508 and for cycles 1 to N1 to the neural network based base-caller 2308 executed by the configurable processor 450. The weight set (cl,(1-N1)) for the central lane tiles 1510 and for cycles 1 to N1 are used to reconfigure the neural network based base-caller 2308.

[0233] At action 3b, the reconfigured neural network-based base caller 2308, running on the configurable processor 450, generates initial representations (*e.g.*, feature maps) from the cluster data (*e.g.*, by processing the cluster data through its spatial and temporal convolution layers), and produces base call classification scores for the plurality of clusters within the central lane tiles 1510 and for the sequencing cycles 1 to N1 based on the initial intermediate representations. In one implementation, the initial base call classification scores are unnormalized, for example, they are not subjected to exponential normalization by a softmax function.

[0234] At action 4b, the configurable processor 450 sends to the data flow logic 451 the base call classification scores for the clusters within the central lane tiles 1510 and for cycles 1 to N1.

At action 5b, the data flow logic 451 provides, to the host processor 2304, the base call classification scores for the clusters within the central lane tiles 1510 and for cycles 1 to N1.

[0235] At action 6b, the host processor 2304 normalizes the unnormalized base call classification scores (*e.g.*, by applying the softmax function), and generates normalized base call classification scores, *i.e.*, base calls, for strands within the clusters of the central lane tiles 1510 and for cycles 1 to N1.

[0236] Thus, base calling operations 1a-6a are specifically for the peripheral lane tiles 1508 and cycles 1 to N1. Similarly, base calling operations 1b-6b are specifically for the central lane tiles 1510 and cycles 1 to N1. The operations of 1a-6a and for 1b-6b are repeated for cycles (N1+1) to N2, and are further repeated for cycles (N2+1) to N, as symbolically illustrated in Fig. 22.

[0237] Referring back to Fig. 7, the illustrated model includes segregated stacks 701, 702, 703, 704, 705. For example, stack 701 receives as input, tile data for the patch from cycle K+2. Stack 702 receives as input, tile data for the patch from cycle K+1. Stack 703 receives as input, tile data for the patch from cycle K. Stack 704 receives as input, tile data for the patch from cycle K-1. Stack 705 receives as input, tile data for the patch from cycle K-2. The layers of the segregated stacks each execute a convolution operation of a kernel including a plurality of filters over the input data for the layer. The output feature set (intermediate data) from each of the stacks 701-705 is provided as input to an inverse hierarchy 720 of temporal combinatorial layers, in which the intermediate data from the multiple cycles are combined.

[0238] Thus, as discussed with respect to Figs. 7, 9, and 11, the stacks 701, ..., 705 perform segregated spatial convolutions. There is no temporal comingling or interaction between input from various cycles within the various stacks 701, ..., 705. Finally, after data processing in the stacks 701, ..., 705, in section 720 there is processing of data from various sequential cycles. The various layers within the stacks 701, ..., 705 are also referred to herein as spatial layers, and weights of kernels of various filters within the stacks 701, ..., 705 are referred to herein as spatial weights. Similarly, the various layers within the section 720 are also referred to herein as temporal layers, and weights of kernels of various filters within the section 720 are also referred to herein as temporal weights. For example, the weights applied during the spatial convolutions 921, 922, 923 in Fig. 9 are the spatial weights, whereas the weights applied during the temporal convolutions 924, 925 in Fig. 9 are the temporal weights.

[0239] **Fig. 23A** illustrates various weights sets for various categories of tiles and for various sensing cycles, individual weight set comprising corresponding spatial weights and corresponding temporal weights. The tile categorization illustrated in Fig. 23A is similar to that discussed with respect to Figs. 15 and 21A. As discussed with respect to Fig. 21A, peripheral

lane tiles 1508 for cycles 1 to N1 are associated with corresponding weight set (pl, 1-N1). As illustrated in Fig. 23A, the weight set (pl, 1-N1) includes corresponding spatial weights (s-pl,(1-N1)) and corresponding temporal weights (t-pl,(1-N1)). The spatial weights (s-pl,(1-N1)) are used to configure the spatial layers of the neural network model, when the neural network model is to process cluster sensor data for peripheral lane tiles 1508 for cycles 1 to N1. The temporal weights (t-pl,(1-N1)) are used to configure the temporal layers of the neural network model, when the neural network model is to process cluster sensor data for peripheral lane tiles 1508 for cycles 1 to N1.

[0240] Similarly, as also discussed with respect to Fig. 21A, peripheral lane tiles 1508 for cycles N1 to N2 are associated with corresponding weight set (pl, N1-N2). As illustrated in Fig. 23A, the weight set (pl, N1-N2) includes corresponding spatial weights (s-pl,(N1-N2)) and corresponding temporal weights (t-pl,(N1-N2)). Various other weight sets of Fig. 23A also similarly have corresponding spatial and temporal weights.

[0241] **Fig. 23B** illustrates various weights sets for various categories of tiles and for various cycles, wherein different weight sets of a specific category of tile includes common spatial weights and different temporal weights. The tile categorization illustrated in Fig. 23A is similar to that discussed with respect to Figs. 15, 21A, 23A. However, unlike Fig. 23A, in Fig. 23B, the weight sets (pl,(1-N1)), (pl,(N1-N2)), and (pl,(N2-N)) for the peripheral lane tiles 1508 have common spatial weights (s-pl). Thus, the same or common spatial weights (s-pl) are used for the peripheral lane tiles 1508 and for each of the subseries of cycles 1 to N1, (N+1) to N2, and (N2+1) to N.

[0242] The weight sets (pl,(1-N1)), (pl,(N1-N2)), and (pl,(N2-N)) have different temporal weights, such as temporal weights (t-pl, (1-N1)), temporal weights (t-pl, (N1-N2)), and temporal weights (t-pl, (N2-N)), respectively.

[0243] Similarly, the weight sets (cl,(1-N1)), (cl,(N1-N2)), and (cl,(N2-N)) for the central lane tiles 1510 have common spatial weights (s-cl). Thus, the same or common spatial weights (s-cl) are used for the central lane tiles 1510 and for each of the subseries of cycles 1 to N1, (N+1) to N2, and (N2+1) to N.

[0244] The weight sets (cl,(1-N1)), (cl,(N1-N2)), and (cl,(N2-N)) have different temporal weights, such as temporal weights (t-cl, (1-N1)), temporal weights (t-cl, (N1-N2)), and temporal weights (t-cl, (N2-N)), respectively.

[0245] In an embodiment and as discussed with respect to in Figs. 17A and 17B, fading, phasing, and/or pre-phasing cause deterioration of sensor data, as sequencing cycles progress. Such deterioration is addressed by the temporal layers (such as layers within block 720 of Fig. 7, or layers 924, 925 of Fig. 9) of the neural network model. Accordingly, in Fig. 23B, the temporal

weights for various subseries of sequencing cycles are trained differently. For example, temporal weights for cycles 1 to N1 and for a given tile category are different from the temporal weights for cycles N1 to N2 for the same tile category. In contrast, as spatial layers (such as layers within blocks 701, ..., 705 of Fig. 7, or layers 921, 922, 923 of Fig. 9) may not significantly cure the deterioration in signal quality, all cycles share common spatial weights for a given tile category, as illustrated in Fig. 23B.

[0246] Thus, when processing sensor data of a specific tile category, say for peripheral lane tiles 1508, the common spatial weights (s-pl) and the temporal weights (t-pl, (1-N1)) of the weight set (pl,(1-N1)) for cycles 1 to N1 are initially loaded in the configurable processor, and the neural network-based base caller 2308 is configured with these spatial and temporal weights. For example, the spatial layers of the neural network-based base caller 2308 are configured with the common spatial weights (s-pl), and the temporal layers of the neural network-based base caller 2308 are configured with the temporal weights (t-pl, (1-N1)). The configured neural network-based base caller 2308 applies the configured spatial and temporal layers on the sensor data for cycles 1 to N1 of the peripheral lane tiles 1508, to produce base call classification data for the cycles 1 to N1 of the peripheral lane tiles 1508.

[0247] Subsequently, before processing sensor data for cycle (N1+1), the temporal weights (t-pl, (N1-N2)) of the weight set (pl,(N1-N2)) are loaded, without loading any corresponding spatial weights of this weight set. The temporal layers of the neural network-based base caller 2308 are configured with the temporal weights (t-pl, (N1-N2)). The neural network-based base caller 2308 then applies the previously configured spatial layers (*e.g.*, which were configured previously with the common spatial weights (s-pl)) and the reconfigured temporal layers (*e.g.*, which are reconfigured with the temporal weights (t-pl, (N1-N2))) on the sensor data for cycles (N1+1) to N2 of the peripheral lane tiles 1508 to produce base call classification data for the cycles (N1+1) to N2 of the peripheral lane tiles 1508.

[0248] Subsequently, before processing sensor data for cycle (N2+1), the temporal weights (t-pl, (N2-N)) of the weight set (pl,(N2-N)) are loaded, without loading any corresponding spatial weights of this weight set. The temporal layers of the neural network-based base caller 2308 are reconfigured with the temporal weights (t-pl, (N2-N)). The neural network-based base caller 2308 then applies the previously configured spatial layers (*e.g.*, which were configured previously with the common spatial weights (s-pl)) and the reconfigured temporal layers (*e.g.*, which are reconfigured with the temporal weights (t-pl, (N2-N))) on the sensor data for cycles (N2+1) to N of the peripheral lane tiles to produce base call classification data for the cycles (N2+1) to N of the peripheral lane tiles.

[0249] The base call classification data for other tile categories (such as the central lane tiles 1510) are produced in a corresponding similar manner, which will be appreciated by those skilled in the art based on the above discussion and the illustration of Fig. 23B.

[0250] Fig. 23C illustrates a system 2300 to select weight sets based on one or more sequencing run parameters 2382. For example, illustrated is a weight set selection logic 2386, which may be executed on the configurable processor 450 and/or the host processor 2304. The weight set selection logic 2386 receives the one or more sequencing run parameters 2382, as well as one or more other weight set selection criteria discussed with respect to Figs. 14-23B. The weight set selection logic 2386 selects a weight set from among a plurality of candidate weight sets 2384a, ..., 2384N, based on the one or more sequencing run parameters 2382 and/or one or more other weight set selection criteria discussed with respect to Figs. 14-23B. In the example of Fig. 23B, the weight set selection logic 2386 selects the weight set 2384b. The selected weight set is then loaded in the configurable processor 450, and used to configure the neural network topology for base calling, as discussed herein.

[0251] The one or more sequencing run parameters 2382 may include one or more appropriate parameters associated with the current sequencing run. For example, reaction components (such as reagents, enzymes, samples, other biomolecules, and buffer solutions) used in the sequencing run can affect the sensor data, and the weight sets can be selected based on the type, parameters, or batch of reaction components used. For example, phasing characteristics (see Fig. 17B) can be based on a reagent pack used for the sequencing run, and can vary based on a type, age, and/or batch of the reagent pack. Thus, the various candidate weight sets can be generated for various types of batches of reaction components, and the weight set selection logic 2386 can select a weight set based on the reaction components used for the current sequencing cycle.

[0252] In another example, the weight set selection logic 2386 can estimate phasing characteristics, and select a weight set based on the phasing characteristics. For example, different weight sets can be generated for different phasing characteristics. Then early in the sequencing run, a phasing parameter can be estimated and used to select a weight set. In yet another example, multiple candidate weight sets can be tried, and the weight set with the lowest error rate (or highest signal to noise ratio) can be selected for the entire sequencing run.

[0253] **Fig. 24** is a block diagram of a base calling system 2400 in accordance with one implementation. The base calling system 2400 may operate to obtain any information or data that relates to at least one of a biological or chemical substance. In some implementations, the base calling system 2400 is a workstation that may be similar to a bench-top device or desktop

computer. For example, a majority (or all) of the systems and components for conducting the desired reactions can be within a common housing 2416.

[0254] In particular implementations, the base calling system 2400 is a nucleic acid sequencing system (or sequencer) configured for various applications, including but not limited to de novo sequencing, resequencing of whole genomes or target genomic regions, and metagenomics. The sequencer may also be used for DNA or RNA analysis. In some implementations, the base calling system 2400 may also be configured to generate reaction sites in a biosensor. For example, the base calling system 2400 may be configured to receive a sample and generate surface attached clusters of clonally amplified nucleic acids derived from the sample. Each cluster may constitute or be part of a reaction site in the biosensor.

[0255] The exemplary base calling system 2400 may include a system receptacle or interface 2412 that is configured to interact with a biosensor 2402 to perform desired reactions within the biosensor 2402. In the following description with respect to Fig. 24, the biosensor 2402 is loaded into the system receptacle 2412. However, it is understood that a cartridge that includes the biosensor 2402 may be inserted into the system receptacle 2412 and in some states the cartridge can be removed temporarily or permanently. As described above, the cartridge may include, among other things, fluidic control and fluidic storage components.

[0256] In particular implementations, the base calling system 2400 is configured to perform a large number of parallel reactions within the biosensor 2402. The biosensor 2402 includes one or more reaction sites where desired reactions can occur. The reaction sites may be, for example, immobilized to a solid surface of the biosensor or immobilized to beads (or other movable substrates) that are located within corresponding reaction chambers of the biosensor. The reaction sites can include, for example, clusters of clonally amplified nucleic acids. The biosensor 2402 may include a solid-state imaging device (*e.g.*, CCD or CMOS imager) and a flow cell mounted thereto. The flow cell may include one or more flow channels that receive a solution from the base calling system 2400 and direct the solution toward the reaction sites. Optionally, the biosensor 2402 can be configured to engage a thermal element for transferring thermal energy into or out of the flow channel.

[0257] The base calling system 2400 may include various components, assemblies, and systems (or sub-systems) that interact with each other to perform a predetermined method or assay protocol for biological or chemical analysis. For example, the base calling system 2400 includes a system controller 2404 that may communicate with the various components, assemblies, and sub-systems of the base calling system 2400 and also the biosensor 2402. For example, in addition to the system receptacle 2412, the base calling system 2400 may also include a fluidic control system 2406 to control the flow of fluid throughout a fluid network of

the base calling system 2400 and the biosensor 2402; a fluidic storage system 2408 that is configured to hold all fluids (*e.g.*, gas or liquids) that may be used by the bioassay system; a temperature control system 2410 that may regulate the temperature of the fluid in the fluid network, the fluidic storage system 2408, and/or the biosensor 2402; and an illumination system 2409 that is configured to illuminate the biosensor 2402. As described above, if a cartridge having the biosensor 2402 is loaded into the system receptacle 2412, the cartridge may also include fluidic control and fluidic storage components.

[0258] Also shown, the base calling system 2400 may include a user interface 2414 that interacts with the user. For example, the user interface 2414 may include a display 2413 to display or request information from a user and a user input device 2415 to receive user inputs. In some implementations, the display 2413 and the user input device 2415 are the same device. For example, the user interface 2414 may include a touch-sensitive display configured to detect the presence of an individual's touch and also identify a location of the touch on the display. However, other user input devices 2415 may be used, such as a mouse, touchpad, keyboard, keypad, handheld scanner, voice-recognition system, motion-recognition system, and the like. As will be discussed in greater detail below, the base calling system 2400 may communicate with various components, including the biosensor 2402 (*e.g.*, in the form of a cartridge), to perform the desired reactions. The base calling system 2400 may also be configured to analyze data obtained from the biosensor to provide a user with desired information.

[0259] The system controller 2404 may include any processor-based or microprocessor-based system, including systems using microcontrollers, Reduced Instruction Set Computers (RISC), Application Specific Integrated Circuits (ASICs), Field Programmable Gate Array (FPGAs), logic circuits, and any other circuit or processor capable of executing functions described herein. The above examples are exemplary only, and are thus not intended to limit in any way the definition and/or meaning of the term system controller. In the exemplary implementation, the system controller 2404 executes a set of instructions that are stored in one or more storage elements, memories, or modules in order to at least one of obtain and analyze detection data. Detection data can include a plurality of sequences of pixel signals, such that a sequence of pixel signals from each of the millions of sensors (or pixels) can be detected over many base calling cycles. Storage elements may be in the form of information sources or physical memory elements within the base calling system 2400.

[0260] The set of instructions may include various commands that instruct the base calling system 2400 or biosensor 2402 to perform specific operations such as the methods and processes of the various implementations described herein. The set of instructions may be in the form of a software program, which may form part of a tangible, non-transitory computer readable medium

or media. As used herein, the terms “software” and “firmware” are interchangeable and include any computer program stored in memory for execution by a computer, including RAM memory, ROM memory, EPROM memory, EEPROM memory, and non-volatile RAM (NVRAM) memory. The above memory types are exemplary only, and are thus not limiting as to the types of memory usable for storage of a computer program.

[0261] The software may be in various forms such as system software or application software. Further, the software may be in the form of a collection of separate programs, or a program module within a larger program or a portion of a program module. The software also may include modular programming in the form of object-oriented programming. After obtaining the detection data, the detection data may be automatically processed by the base calling system 2400, processed in response to user inputs, or processed in response to a request made by another processing machine (e.g., a remote request through a communication link). In the illustrated implementation, the system controller 2404 includes an analysis module 2538 (illustrated in Fig. 25). In other implementations, system controller 2404 does not include the analysis module 2538 and instead has access to the analysis module 2538 (e.g., the analysis module 2538 may be separately hosted on cloud).

[0262] The system controller 2404 may be connected to the biosensor 2402 and the other components of the base calling system 2400 via communication links. The system controller 2404 may also be communicatively connected to off-site systems or servers. The communication links may be hardwired, corded, or wireless. The system controller 2404 may receive user inputs or commands, from the user interface 2414 and the user input device 2415.

[0263] The fluidic control system 2406 includes a fluid network and is configured to direct and regulate the flow of one or more fluids through the fluid network. The fluid network may be in fluid communication with the biosensor 2402 and the fluidic storage system 2408. For example, select fluids may be drawn from the fluidic storage system 2408 and directed to the biosensor 2402 in a controlled manner, or the fluids may be drawn from the biosensor 2402 and directed toward, for example, a waste reservoir in the fluidic storage system 2408. Although not shown, the fluidic control system 2406 may include flow sensors that detect a flow rate or pressure of the fluids within the fluid network. The sensors may communicate with the system controller 2404.

[0264] The temperature control system 2410 is configured to regulate the temperature of fluids at different regions of the fluid network, the fluidic storage system 2408, and/or the biosensor 2402. For example, the temperature control system 2410 may include a thermocycler that interfaces with the biosensor 2402 and controls the temperature of the fluid that flows along the reaction sites in the biosensor 2402. The temperature control system 2410 may also regulate

the temperature of solid elements or components of the base calling system 2400 or the biosensor 2402. Although not shown, the temperature control system 2410 may include sensors to detect the temperature of the fluid or other components. The sensors may communicate with the system controller 2404.

[0265] The fluidic storage system 2408 is in fluid communication with the biosensor 2402 and may store various reaction components or reactants that are used to conduct the desired reactions therein. The fluidic storage system 2408 may also store fluids for washing or cleaning the fluid network and biosensor 2402 and for diluting the reactants. For example, the fluid storage system 2408 may include various reservoirs to store samples, reagents, enzymes, other biomolecules, buffer solutions, aqueous, and non-polar solutions, and the like. Furthermore, the fluidic storage system 2408 may also include waste reservoirs for receiving waste products from the biosensor 2402. In implementations that include a cartridge, the cartridge may include one or more of a fluid storage system, fluidic control system or temperature control system.

Accordingly, one or more of the components set forth herein as relating to those systems can be contained within a cartridge housing. For example, a cartridge can have various reservoirs to store samples, reagents, enzymes, other biomolecules, buffer solutions, aqueous, and non-polar solutions, waste, and the like. As such, one or more of a fluid storage system, fluidic control system or temperature control system can be removably engaged with a bioassay system via a cartridge or other biosensor.

[0266] The illumination system 2409 may include a light source (*e.g.*, one or more LEDs) and a plurality of optical components to illuminate the biosensor. Examples of light sources may include lasers, arc lamps, LEDs, or laser diodes. The optical components may be, for example, reflectors, dichroics, beam splitters, collimators, lenses, filters, wedges, prisms, mirrors, detectors, and the like. In implementations that use an illumination system, the illumination system 2409 may be configured to direct an excitation light to reaction sites. As one example, fluorophores may be excited by green wavelengths of light, as such the wavelength of the excitation light may be approximately 532 nm. In one implementation, the illumination system 2409 is configured to produce illumination that is parallel to a surface normal of a surface of the biosensor 2402. In another implementation, the illumination system 2409 is configured to produce illumination that is off-angle relative to the surface normal of the surface of the biosensor 2402. In yet another implementation, the illumination system 2409 is configured to produce illumination that has plural angles, including some parallel illumination and some off-angle illumination.

[0267] The system receptacle or interface 2412 is configured to engage the biosensor 2402 in at least one of a mechanical, electrical, and fluidic manner. The system receptacle 2412 may hold

the biosensor 2402 in a desired orientation to facilitate the flow of fluid through the biosensor 2402. The system receptacle 2412 may also include electrical contacts that are configured to engage the biosensor 2402 so that the base calling system 2400 may communicate with the biosensor 2402 and/or provide power to the biosensor 2402. Furthermore, the system receptacle 2412 may include fluidic ports (*e.g.*, nozzles) that are configured to engage the biosensor 2402. In some implementations, the biosensor 2402 is removably coupled to the system receptacle 2412 in a mechanical manner, in an electrical manner, and also in a fluidic manner.

[0268] In addition, the base calling system 2400 may communicate remotely with other systems or networks or with other bioassay systems 2400. Detection data obtained by the bioassay system(s) 2400 may be stored in a remote database.

[0269] **Fig. 25** is a block diagram of the system controller 2404 that can be used in the system of Fig. 24. In one implementation, the system controller 2404 includes one or more processors or modules that can communicate with one another. Each of the processors or modules may include an algorithm (*e.g.*, instructions stored on a tangible and/or non-transitory computer readable storage medium) or sub-algorithms to perform particular processes. The system controller 2404 is illustrated conceptually as a collection of modules, but may be implemented utilizing any combination of dedicated hardware boards, DSPs, processors, etc. Alternatively, the system controller 2404 may be implemented utilizing an off-the-shelf PC with a single processor or multiple processors, with the functional operations distributed between the processors. As a further option, the modules described below may be implemented utilizing a hybrid configuration in which certain modular functions are performed utilizing dedicated hardware, while the remaining modular functions are performed utilizing an off-the-shelf PC and the like. The modules also may be implemented as software modules within a processing unit.

[0270] During operation, a communication port 2520 may transmit information (*e.g.*, commands) to or receive information (*e.g.*, data) from the biosensor 2402 (Fig. 24) and/or the sub-systems 2406, 2408, 2410 (Fig. 24). In implementations, the communication port 2520 may output a plurality of sequences of pixel signals. A communication port 2520 may receive user input from the user interface 2414 (Fig. 24) and transmit data or information to the user interface 2414. Data from the biosensor 2402 or sub-systems 2406, 2408, 2410 may be processed by the system controller 2404 in real-time during a bioassay session. Additionally, or alternatively, data may be stored temporarily in a system memory during a bioassay session and processed in slower than real-time or off-line operation.

[0271] As shown in Fig. 25, the system controller 2404 may include a plurality of modules 2531-2539 that communicate with a main control module 2530. The main control module 2530 may communicate with the user interface 2414 (Fig. 24). Although the modules 2531-2539 are

shown as communicating directly with the main control module 2530, the modules 2531-2539 may also communicate directly with each other, the user interface 2414, and the biosensor 2402. Also, the modules 2531-2539 may communicate with the main control module 2530 through the other modules.

[0272] The plurality of modules 2531-2539 include system modules 2531-2533, 2539 that communicate with the sub-systems 2406, 2408, 2410, and 2409, respectively. The fluidic control module 2531 may communicate with the fluidic control system 2406 to control the valves and flow sensors of the fluid network for controlling the flow of one or more fluids through the fluid network. The fluidic storage module 2532 may notify the user when fluids are low or when the waste reservoir is at or near capacity. The fluidic storage module 2532 may also communicate with the temperature control module 2533 so that the fluids may be stored at a desired temperature. The illumination module 2539 may communicate with the illumination system 2409 to illuminate the reaction sites at designated times during a protocol, such as after the desired reactions (*e.g.*, binding events) have occurred. In some implementations, the illumination module 2539 may communicate with the illumination system 2409 to illuminate the reaction sites at designated angles.

[0273] The plurality of modules 2531-2539 may also include a device module 2534 that communicates with the biosensor 2402 and an identification module 2535 that determines identification information relating to the biosensor 2402. The device module 2534 may, for example, communicate with the system receptacle 2412 to confirm that the biosensor has established an electrical and fluidic connection with the base calling system 2400. The identification module 2535 may receive signals that identify the biosensor 2402. The identification module 2535 may use the identity of the biosensor 2402 to provide other information to the user. For example, the identification module 2535 may determine and then display a lot number, a date of manufacture, or a protocol that is recommended to be run with the biosensor 2402.

[0274] The plurality of modules 2531-2539 also includes an analysis module 2538 (also called signal processing module or signal processor) that receives and analyzes the signal data (*e.g.*, image data) from the biosensor 2402. Analysis module 2538 includes memory (*e.g.*, RAM or Flash) to store detection data. Detection data can include a plurality of sequences of pixel signals, such that a sequence of pixel signals from each of the millions of sensors (or pixels) can be detected over many base calling cycles. The signal data may be stored for subsequent analysis or may be transmitted to the user interface 2414 to display desired information to the user. In some implementations, the signal data may be processed by the solid-state imager (*e.g.*, CMOS image sensor) before the analysis module 2538 receives the signal data.

[0275] The analysis module 2538 is configured to obtain image data from the light detectors at each of a plurality of sequencing cycles. The image data is derived from the emission signals detected by the light detectors and process the image data for each of the plurality of sequencing cycles through a neural network (*e.g.*, a neural network-based template generator 2548, a neural network-based base caller 2558 (*e.g.*, see Figs. 7, 9, and 10), and/or a neural network-based quality scorer 2568) and produce a base call for at least some of the analytes at each of the plurality of sequencing cycle.

[0276] Protocol modules 2536 and 2537 communicate with the main control module 2530 to control the operation of the sub-systems 2406, 2408, and 2410 when conducting predetermined assay protocols. The protocol modules 2536 and 2537 may include sets of instructions for instructing the base calling system 2400 to perform specific operations pursuant to predetermined protocols. As shown, the protocol module may be a sequencing-by-synthesis (SBS) module 2536 that is configured to issue various commands for performing sequencing-by-synthesis processes. In SBS, extension of a nucleic acid primer along a nucleic acid template is monitored to determine the sequence of nucleotides in the template. The underlying chemical process can be polymerization (*e.g.*, as catalyzed by a polymerase enzyme) or ligation (*e.g.*, catalyzed by a ligase enzyme). In a particular polymerase-based SBS implementation, fluorescently labeled nucleotides are added to a primer (thereby extending the primer) in a template dependent fashion such that detection of the order and type of nucleotides added to the primer can be used to determine the sequence of the template. For example, to initiate a first SBS cycle, commands can be given to deliver one or more labeled nucleotides, DNA polymerase, etc., into/through a flow cell that houses an array of nucleic acid templates. The nucleic acid templates may be located at corresponding reaction sites. Those reaction sites where primer extension causes a labeled nucleotide to be incorporated can be detected through an imaging event. During an imaging event, the illumination system 2409 may provide an excitation light to the reaction sites. Optionally, the nucleotides can further include a reversible termination property that terminates further primer extension once a nucleotide has been added to a primer. For example, a nucleotide analog having a reversible terminator moiety can be added to a primer such that subsequent extension cannot occur until a deblocking agent is delivered to remove the moiety. Thus, for implementations that use reversible termination a command can be given to deliver a deblocking reagent to the flow cell (before or after detection occurs). One or more commands can be given to effect wash(es) between the various delivery steps. The cycle can then be repeated *n* times to extend the primer by *n* nucleotides, thereby detecting a sequence of length *n*. Exemplary sequencing techniques are described, for example, in Bentley et al., Nature 456:53-59 (2008); WO 04/018497; US 7,057,026; WO 91/06678; WO 07/123744; US

7,329,492; US 7,211,414; US 7,315,019; and US 7,405,281, each of which is incorporated herein by reference.

[0277] For the nucleotide delivery step of an SBS cycle, either a single type of nucleotide can be delivered at a time, or multiple different nucleotide types (*e.g.*, A, C, T and G together) can be delivered. For a nucleotide delivery configuration where only a single type of nucleotide is present at a time, the different nucleotides need not have distinct labels since they can be distinguished based on temporal separation inherent in the individualized delivery. Accordingly, a sequencing method or apparatus can use single color detection. For example, an excitation source need only provide excitation at a single wavelength or in a single range of wavelengths. For a nucleotide delivery configuration where delivery results in multiple different nucleotides being present in the flow cell at one time, sites that incorporate different nucleotide types can be distinguished based on different fluorescent labels that are attached to respective nucleotide types in the mixture. For example, four different nucleotides can be used, each having one of four different fluorophores. In one implementation, the four different fluorophores can be distinguished using excitation in four different regions of the spectrum. For example, four different excitation radiation sources can be used. Alternatively, fewer than four different excitation sources can be used, but optical filtration of the excitation radiation from a single source can be used to produce different ranges of excitation radiation at the flow cell.

[0278] In some implementations, fewer than four different colors can be detected in a mixture having four different nucleotides. For example, pairs of nucleotides can be detected at the same wavelength, but distinguished based on a difference in intensity for one member of the pair compared to the other, or based on a change to one member of the pair (*e.g.*, via chemical modification, photochemical modification or physical modification) that causes apparent signal to appear or disappear compared to the signal detected for the other member of the pair. Exemplary apparatus and methods for distinguishing four different nucleotides using detection of fewer than four colors are described for example in US Pat. App. Ser. Nos. 61/538,294 and 61/619,878, which are incorporated herein by reference in their entireties. U.S. Application No. 13/624,200, which was filed on September 21, 2012, is also incorporated by reference in its entirety.

[0279] The plurality of protocol modules may also include a sample-preparation (or generation) module 2537 that is configured to issue commands to the fluidic control system 2406 and the temperature control system 2410 for amplifying a product within the biosensor 2402. For example, the biosensor 2402 may be engaged to the base calling system 2400. The amplification module 2537 may issue instructions to the fluidic control system 2406 to deliver necessary amplification components to reaction chambers within the biosensor 2402. In other

implementations, the reaction sites may already contain some components for amplification, such as the template DNA and/or primers. After delivering the amplification components to the reaction chambers, the amplification module 2537 may instruct the temperature control system 2410 to cycle through different temperature stages according to known amplification protocols. In some implementations, the amplification and/or nucleotide incorporation is performed isothermally.

[0280] The SBS module 2536 may issue commands to perform bridge PCR where clusters of clonal amplicons are formed on localized areas within a channel of a flow cell. After generating the amplicons through bridge PCR, the amplicons may be “linearized” to make single stranded template DNA, or sstDNA, and a sequencing primer may be hybridized to a universal sequence that flanks a region of interest. For example, a reversible terminator-based sequencing by synthesis method can be used as set forth above or as follows.

[0281] Each base calling or sequencing cycle can extend an sstDNA by a single base which can be accomplished for example by using a modified DNA polymerase and a mixture of four types of nucleotides. The different types of nucleotides can have unique fluorescent labels, and each nucleotide can further have a reversible terminator that allows only a single-base incorporation to occur in each cycle. After a single base is added to the sstDNA, excitation light may be incident upon the reaction sites and fluorescent emissions may be detected. After detection, the fluorescent label and the terminator may be chemically cleaved from the sstDNA. Another similar base calling or sequencing cycle may follow. In such a sequencing protocol, the SBS module 2536 may instruct the fluidic control system 2406 to direct a flow of reagent and enzyme solutions through the biosensor 2402. Exemplary reversible terminator-based SBS methods which can be utilized with the apparatus and methods set forth herein are described in US Patent Application Publication No. 2007/0166705 A1, US Patent Application Publication No. 2006/0188901 A1, US Patent No. 7,057,026, US Patent Application Publication No. 2006/0240439 A1, US Patent Application Publication No. 2006/02814714709 A1, PCT Publication No. WO 05/065814, PCT Publication No. WO 06/064199, each of which is incorporated herein by reference in its entirety. Exemplary reagents for reversible terminator-based SBS are described in US 7,541,444; US 7,057,026; US 7,427,673; US 7,566,537; and US 7,592,435, each of which is incorporated herein by reference in its entirety.

[0282] In some implementations, the amplification and SBS modules may operate in a single assay protocol where, for example, template nucleic acid is amplified and subsequently sequenced within the same cartridge.

[0283] The base calling system 2400 may also allow the user to reconfigure an assay protocol. For example, the base calling system 2400 may offer options to the user through the

user interface 2414 for modifying the determined protocol. For example, if it is determined that the biosensor 2402 is to be used for amplification, the base calling system 2400 may request a temperature for the annealing cycle. Furthermore, the base calling system 2400 may issue warnings to a user if a user has provided user inputs that are generally not acceptable for the selected assay protocol.

[0284] In implementations, the biosensor 2402 includes millions of sensors (or pixels), each of which generates a plurality of sequences of pixel signals over successive base calling cycles. The analysis module 2538 detects the plurality of sequences of pixel signals and attributes them to corresponding sensors (or pixels) in accordance to the row-wise and/or column-wise location of the sensors on an array of sensors.

[0285] Each sensor in the array of sensors can produce sensor data for a tile of the flow cell, where a tile is an area on the flow cell at which clusters of genetic material are disposed during the based calling operation. The sensor data can comprise image data in an array of pixels. For a given cycle, the sensor data can include more than one image, producing multiple features per pixel as the tile data.

[0286] **Fig. 26** is a simplified block diagram of a computer 2600 system that can be used to implement the technology disclosed. Computer system 2600 includes at least one central processing unit (CPU) 2672 that communicates with a number of peripheral devices via bus subsystem 2655. These peripheral devices can include a storage subsystem 2610 including, for example, memory devices and a file storage subsystem 2636, user interface input devices 2638, user interface output devices 2676, and a network interface subsystem 2674. The input and output devices allow user interaction with computer system 2600. Network interface subsystem 2674 provides an interface to outside networks, including an interface to corresponding interface devices in other computer systems.

[0287] User interface input devices 2638 can include a keyboard; pointing devices such as a mouse, trackball, touchpad, or graphics tablet; a scanner; a touch screen incorporated into the display; audio input devices such as voice recognition systems and microphones; and other types of input devices. In general, use of the term “input device” is intended to include all possible types of devices and ways to input information into computer system 2600.

[0288] User interface output devices 2676 can include a display subsystem, a printer, a fax machine, or non-visual displays such as audio output devices. The display subsystem can include an LED display, a cathode ray tube (CRT), a flat-panel device such as a liquid crystal display (LCD), a projection device, or some other mechanism for creating a visible image. The display subsystem can also provide a non-visual display such as audio output devices. In general, use of

the term “output device” is intended to include all possible types of devices and ways to output information from computer system 2600 to the user or to another machine or computer system.

[0289] Storage subsystem 2610 stores programming and data constructs that provide the functionality of some or all of the modules and methods described herein. These software modules are generally executed by deep learning processors 2678.

[0290] In one implementation, the neural networks are implemented using deep learning processors 2678 can be configurable and reconfigurable processors, field-programmable gate arrays (FPGAs), application-specific integrated circuits (ASICs), and/or coarse-grained reconfigurable architectures (CGRAs) and graphics processing units (GPUs) other configured devices. Deep learning processors 2678 can be hosted by a deep learning cloud platform such as Google Cloud Platform™, Xilinx™, and Cirrascale™. Examples of deep learning processors 14978 include Google’s Tensor Processing Unit (TPU)™, rackmount solutions like GX4 Rackmount Series™, GX149 Rackmount Series™, NVIDIA DGX-1™, Microsoft’ Stratix V FPGA™, Graphcore’s Intelligent Processor Unit (IPU)™, Qualcomm’s Zeroth Platform™ with Snapdragon processors™, NVIDIA’s Volta™, NVIDIA’s DRIVE PX™, NVIDIA’s JETSON TX1/TX2 MODULE™, Intel’s Nirvana™, Movidius VPU™, Fujitsu DPI™, ARM’s DynamicIQ™, IBM TrueNorth™, and others.

[0291] Memory subsystem 2622 used in the storage subsystem 2610 can include a number of memories including a main random access memory (RAM) 2634 for storage of instructions and data during program execution and a read only memory (ROM) 2632 in which fixed instructions are stored. A file storage subsystem 2636 can provide persistent storage for program and data files, and can include a hard disk drive, a floppy disk drive along with associated removable media, a CD-ROM drive, an optical drive, or removable media cartridges. The modules implementing the functionality of certain implementations can be stored by file storage subsystem 2636 in the storage subsystem 2610, or in other machines accessible by the processor.

[0292] Bus subsystem 2655 provides a mechanism for letting the various components and subsystems of computer system 2600 communicate with each other as intended. Although bus subsystem 2655 is shown schematically as a single bus, alternative implementations of the bus subsystem can use multiple busses.

[0293] Computer system 2600 itself can be of varying types including a personal computer, a portable computer, a workstation, a computer terminal, a network computer, a television, a mainframe, a server farm, a widely-distributed set of loosely networked computers, or any other data processing system or user device. Due to the ever-changing nature of computers and networks, the description of computer system 2600 depicted in Fig. 26 is intended only as a specific example for purposes of illustrating the preferred implementations of the present

invention. Many other configurations of computer system 2600 are possible having more or less components than the computer system depicted in Fig. 26.

[0294] We claim as follows: