

## Downsampling Genomic Sequence Data

### Background of the Invention

[0001] The present invention relates generally to the field of genomic sequence data, and more particularly to downsampling deoxyribonucleic acid (DNA) sequence data.

[0002] Genomes are sequenced every day, wherein a single human genome may take up to 100 gigabytes of storage space, thus, storage needs for genomic sequence data will grow from gigabytes to petabytes to exabytes. Moreover, for every 3 billion bases of the human genome sequence, a 30-fold increase in data (approximately 100 gigabases) must be collected because of errors in sequencing, base calling, and genome alignment. This means that as much as 2–40 exabytes of storage capacity will be needed by the year 2025 just for the human genomes. In digital signal processing, downsampling, compression, and decimation are terms associated with the process of resampling in a multi-rate digital signal processing system. Both downsampling and decimation can be synonymous with compression, or they can describe an entire process of bandwidth reduction (filtering) and sample-rate reduction. When the process is performed on a sequence of samples of a particular signal or other continuous functions, downsampling produces an approximation of the sequence that would have been obtained by sampling the signal at a lower rate (or density, as in the case of a photograph).

[0003] The advent of massively parallel high-throughput sequencing technology known as next generation sequencing (NGS) has revolutionized the field of biological sciences. The advent of NGS has led to an increase in the amount and availability of DNA sequence data. The ultra-high scale and efficient NGS sequencing machines can produce billions of short DNA reads in excess of a few terabytes of data in a single run. The short DNA reads are stored in FASTQ files (plain text files in gigabytes), which are used for a wide range of applications such as gene annotations, expression studies, personalized treatment and precision medicine, wherein a FASTQ file is a file in FASTQ format. FASTQ format is a text-based format for storing both a biological sequence (usually nucleotide sequence) and its corresponding quality scores. Both the sequence letter and quality score are each encoded with a single American Standard Code for Information Interchange (ASCII) character for brevity.

## SUMMARY

[0004] Embodiments of the present invention disclose a method, a computer program product, and a system for automatically downsampling DNA sequence data using variational autoencoders and preserving genomic integrity of an original file, the method comprising: executing, by an encoder, bootstrapping on genomic sequence data to produce resamples; assessing, by the encoder, unrepresentativeness and self-inconsistency of the resamples and selecting a representative resample according to the assessment; building, by a modified encoder, vector representations from genotype likelihoods based on the selected representative sample; integrating, by an analytics engine, mapping positional information and the genotype likelihoods to identify an optimum vector representation of a resample; and decoding, by a modified decoder, the identified optimum vector representation of the resample to obtain a down-sampled read file that resembles and maintains the genomic integrity of the original file.

## BRIEF DESCRIPTION OF THE DRAWINGS

[0005] FIG. 1 is a functional block diagram illustrating a distributed data processing environment, in accordance with an embodiment of the present invention;

[0006] FIG. 2A illustrates a block diagram depicting a downsampling component, executing on a server computer, within distributed data processing environment of FIG. 1, for downsampling DNA sequence data using variational autoencoders, in accordance with an embodiment of the present invention;

[0007] FIG. 2B is an example of the overview of the bootstrapping procedure, in accordance with an embodiment of the present invention;

[0008] FIG. 2C illustrates a block diagram depicting an overview of biological information generation, within distributed data processing environment of FIG. 1, for downsampling DNA sequence data using variational autoencoders, in accordance with an embodiment of the present invention

[0009] FIG. 3 illustrates operational steps of the downsampling component, on a computing device within the distributed data processing environment of FIG. 1, for automatically downsampling DNA sequence data using variational autoencoders, in accordance with an embodiment of the present invention; and

[0010] FIG. 4 depicts a block diagram of components of the server computer executing the downsampling component within the distributed data processing environment of FIG. 1, in accordance with an embodiment of the present invention.

## DETAILED DESCRIPTION

[0011] Embodiments of the present invention recognize that there are issues with the current methods and systems for genomic data analysis and integrating and archiving DNA sequence data. For example, the high-throughput genome sequencing inherently produces files containing millions of reads per sample, files requiring large computing resources for storage and processing, and samples that need to be effectively summarized (e.g. downsampling) to optimize computational efficiency. Embodiments of the present invention recognize that current methods of downsampling do not maintain the genomic integrity of the original sample.

[0012] It is known in the art that the surge of genomic data poses huge challenges in analyzing, integrating, sharing and archiving DNA sequence data. Embodiments of the present invention solve the problems stated above by presenting a system and method for down-sampling billions of reads in a FASTQ file to reduce the file size and amount of DNA reads while preserving the integrity of the original information and avoiding loss of important genomic information, wherein the loss of information is mainly attributed to the loss of genomic variation such as structural variation that is encoded in a plethora of reads, thus, enabling more effective and efficient downstream processing and analysis of genomic sequence data.

[0013] Additionally, embodiments of the present invention summarize genomic information using a composite representation. Embodiments of the present invention take into account the genome sequence data as a whole and transform the genomic information (e.g., the genome sequence data as a whole) into a representation (e.g., numerical vector) that keeps the original biological information intact, wherein the input data is from DNA sequencing for multiple reads (e.g., analyzes, identifies, and/or interprets transformations for a plurality of reads). In various embodiments of the present invention, the biological information that is mostly captured and encoded into reads are the different types of variations known in the art, such as insertion-deletion mutations (indels), single-nucleotide polymorphisms (SNPs), and/or copy-number variations (cnvs) that can be used to explain different phenotypes. Furthermore, embodiments of the present invention utilize shallow sampling using a bootstrapping approach to calculate a minimum value of representativeness and consistence. In some embodiments of the present invention, the present invention is not focused on single-nucleotide polymorphism (SNP) calling, but rather genotype likelihood estimation at multiple positions within a read. Embodiments of the

present invention may use a graphical Poisson distribution to estimate genotype likelihoods. In various embodiments of the present invention, the present invention is a downsampling method and, in some instances, is alignment free. Embodiments of the present invention recognize that an alignment works when a user (i.e., researcher) possesses a reference genome and in most cases the reference genome is incomplete resulting in the alignment being dependent on the reference genome (i.e., the alignment is only as good as your reference). In various embodiments of the present invention, the present invention is focused on genotype likelihood estimation at multiple positions within a read to use as prior information without relying on variant calling techniques.

**[0014]** Implementation of embodiments of the invention may take a variety of forms, and exemplary implementation details are discussed subsequently with reference to the Figures (i.e., FIG. 1 – FIG. 4).

**[0015]** FIG. 1 is a functional block diagram illustrating a distributed data processing environment, generally designated 100, in accordance with one embodiment of the present invention. The term “distributed” as used in this specification describes a computer system that includes multiple, physically distinct devices that operate together as a single computer system. FIG. 1 provides only an illustration of one implementation and does not imply any limitations with regard to the environments in which different embodiments may be implemented. Many modifications to the depicted environment may be made by those skilled in the art without departing from the scope of the invention as recited by the claims. Distributed data processing environment 100 includes computing device 110 and server computer 120 interconnected over network 130.

**[0016]** Network 130 may be, for example, a storage area network (SAN), a telecommunications network, a local area network (LAN), a wide area network (WAN), such as the Internet, a wireless technology for exchanging data over short distances (using short-wavelength ultra-high frequency (UHF) radio waves in the industrial, scientific and medical (ISM) band from 2.4 to 2.485 GHz from fixed and mobile devices, and building personal area networks (PANs) or a combination of the three), and may include wired, wireless, or fiber optic connections. Network 130 may include one or more wired and/or wireless networks that may receive and transmit data, voice, and/or video signals, including multimedia signals that include

voice, data, text and/or video data. In general, network 130 may be any combination of connections and protocols that will support communications between computing device 110 and server computer 120, and any other computing devices and/or storage devices (not shown in FIG. 1) within distributed data processing environment 100.

**[0017]** In some embodiments of the present invention, computing device 110 may be, but is not limited to, a standalone device, a client, a server, a laptop computer, a tablet computer, a netbook computer, a personal computer (PC), a smart phone, a desktop computer, a smart television, a smart watch, a radio, a stereo system, a cloud based service (e.g., a cognitive cloud based service), AR glasses, a virtual reality headset, any HUD known in the art, and/or any programmable electronic computing device capable of communicating with various components and devices within distributed data processing environment 100, via network 130 or any combination therein. In general, computing device 110 may be representative of any programmable computing device or a combination of programmable computing devices capable of executing machine-readable program instructions and communicating with users of other computing devices via network 130 and/or capable of executing machine-readable program instructions and communicating with server computer 120. In some embodiments computing device 110 may represent a plurality of computing devices.

**[0018]** In some embodiments of the present invention, computing device 110 may represent any programmable electronic computing device or combination of programmable electronic computing devices capable of executing machine readable program instructions, manipulating executable machine-readable instructions, and communicating with server computer 120 and other computing devices (not shown) within distributed data processing environment 100 via a network, such as network 130. Computing device 110 may include an instance of user interface (interface) 106, and local storage 104. In various embodiments, not depicted in FIG. 1, computing device 110 may have a plurality of user interfaces. In other embodiments, not depicted in FIG. 1, distributed data processing environment 100 may comprise a plurality of computing devices, plurality of server computers, and/or one a plurality of networks. Computing device 110 may include internal and external hardware components, as depicted, and described in further detail with respect to FIG. 4.

**[0019]** User interface (interface) 106 provides an interface to predictive component 140. Computing device 110, via user interface 106, may enable a user and/or a client to interact with component 140 and/or server computer 120 in various ways, such as sending program instructions, receiving program instructions, sending and/or receiving messages, updating data, sending data, inputting data, editing data, collecting data, and/or receiving data. In one embodiment, interface 106 may be a graphical user interface (GUI) or a web user interface (WUI) and may display at least text, documents, web browser windows, user options, application interfaces, and instructions for operation. interface 106 may include data (such as graphic, text, and sound) presented to a user and control sequences the user employs to control operations. In another embodiment, interface 106 may be a mobile application software providing an interface between a user of computing device 110 and server computer 120. Mobile application software, or an “app,” may be designed to run on smart phones, tablet computers and other computing devices. In an embodiment, interface 106 may enable the user of computing device 110 to at least send data, input data, edit data (annotations), collect data and/or receive data.

**[0020]** Server computer 120 may be a standalone computing device, a management server, a web server, a mobile computing device, one or more client servers, or any other electronic device or computing system capable of receiving, sending, and processing data. In other embodiments, server computer 120 may represent a server computing system utilizing multiple computers such as, but not limited to, a server system, such as in a cloud computing environment. In another embodiment, server computer 120 may represent a computing system utilizing clustered computers and components (e.g., database server computers, application server computers, etc.) that act as a single pool of seamless resources when accessed within distributed data processing environment 100. Server computer 120 may include internal and external hardware components, as depicted, and described in further detail with respect to FIG. 4. In some embodiments server computer 120 may represent a plurality of server computers.

**[0021]** Each of shared storage 124 and local storage 104 may be a data/knowledge repository and/or a database that may be written and/or read by one or a combination of component 140, server computer 120 and computing device 110. In the depicted embodiment, shared storage 124 resides on server computer 120 and local storage 104 resides on computing device 110. In another embodiment, shared storage 124 and/or local storage 104 may reside elsewhere within

distributed data processing environment 100, provided that each may access and is accessible by computing device 110 and server computer 120. Shared storage 124 and/or local storage 104 may each be implemented with any type of storage device capable of storing data and configuration files that may be accessed and utilized by server computer 120, such as, but not limited to, a database server, a hard disk drive, or a flash memory.

**[0022]** In some embodiments of the present invention, shared storage 124 and/or local storage 104 may each be a hard drive, a memory card, a computer output to laser disc (cold storage), and/or any form of data storage known in the art. In some embodiments, shared storage 124 and/or local storage 104 may each be one or more cloud storage systems and/or databases linked to a cloud network. In various embodiments, shared storage 124 and/or local storage 104 may access, store, and/or house user data, physical room data, and meeting data, and/or data shared throughout distributed data processing environment 100.

**[0023]** In the depicted embodiment, downsampling component (component) 140 is executed on server computer 120. In other embodiments, downsampling component 140 may be executed on computing device 110. In various embodiments of the present invention, not depicted in FIG. 1, downsampling component 140 may execute on a plurality of server computers 120 and/or on a plurality of computing devices 110. In some embodiments, component 140 may be located and/or executed anywhere within distributed data processing environment 100 as long as downsampling component 140 is connected to and/or communicates with computing device 110 and/or server computer 120, via network 130. In the depicted embodiment, encoder 142, decoder 144, and analytics engine 146 are each located and executed on downsampling component 140. In various embodiments, not depicted in FIG. 1, encoder 142, decoder 144, and analytics engine 146 may execute anywhere within distributed data processing environment 100 as long as encoder 142, decoder 144, and/or analytics engine 146 are each connected to and/or communicate with computing device 110 and/or server computer 120, via network 130.

**[0024]** In various embodiments of the present invention, encoder 142 may be one or more of: modified encoders, audio encoders, simple encoders, priority encoders, compressors, rotary encoders, linear encoders, absolute encoders, incremental encoders, and/or any type of encoder known in the art. In various embodiments of the present invention, decoder 144 may be one or more of: audio decoders, binary decoders, decompression decoders, instruction decoders,

quadrature decoders, video decoders, and/or any other decoder known in the art. Analytics engine 146 may be one or more analytics engines as they are understood in the art.

[0025] In various embodiments, downsampling component 140, via encoder 142, utilizes a probabilistic resampling framework to select a subset of read fragments from an entire file. The encoder (e.g., encoder 142), performs optimization of bootstrapping to produce a set of resamples for a given N size (e.g., N = 1 million reads that may have a total of 100 resamples). In various embodiments of the present invention, downsampling component 140, via encoder 142, selects a subset of encoded read fragments from one or more resamples after satisfying one or more predetermined criteria. In various embodiments of the present invention, decoder 144 may be used to produce a final set of read fragments that are representative of the original FASTQ file. In various embodiments of the present invention, decoder 144 is used to generate the final read fragments that may be placed in the final resample. In various embodiments of the present invention, decoder 144 may be used to produce  $x'$  (i.e. the final representative resample). Decoder 144 may reconstruct one or more of the vector representations that encode the reads as they move through the pipeline to generate the actual read/DNA fragments.

[0026] In various embodiments of the present invention, downsampling component 140 consists of two steps: first, an encoder (e.g., encoder 142) retrieves and/or receives genomic sequence data and performs bootstrapping to produce resamples, wherein unrepresentativeness and self-inconsistency of the produced resamples are assessed, where the variation is minimized, and wherein the most representative resample is selected; second, a decoder (e.g., decoder 144) receives as an input the most representative sample (e.g., the selected most representative resample) and produces a reconstruction of the original input reads. In various embodiments of the present invention, downsampling component 140 removes duplicate reads from a FASTQ file to reduce redundancy of genomic information. In various embodiments of the present invention, downsampling component 140 may process a FASTQ file containing millions and/or billions of reads by tagging one or more reads (e.g., all the reads) with a unique identifier and then mapping them to a sample reference sequence. In various embodiments of the present invention, downsampling component 140, via encoder 142, may encode positional information and mapping and quality scores for each read fragment. In various embodiments of the present invention, a statistical model is trained to estimate genotype likelihoods at each position and encode this information for each read fragment, via encoder 142.

[0027] In various embodiments of the present invention, downsampling component 140 may, without compromising (i.e., while maintaining) the genomic integrity of the original file, resample DNA fragments generated from massively parallel sequencing. For example, the genomic integrity refers to a genomic variation that results in different phenotypes and is encoded in different reads (i.e., a plurality or plethora of reads). In this particular example, in order to maintain the genomic integrity downsampling component 140 executes downsampling on one or more of the plurality of reads, wherein downsampling component 140 retains a majority or predetermined amounts of reads.

[0028] In various embodiments of the present invention, downsampling component 140, via encoder 142, may retrieve and/or receive genomic sequence data (e.g., DNA sequence data) from one or more users, one or more databases, and/or from one or more search engines and perform bootstrapping on the retrieve and/or receive genomic sequence data to produce resamples. In various embodiments of the present invention, downsampling component 140, via encoder 142, may perform probabilistic resampling to obtain a set of resamples containing a plurality of reads. In various embodiments of the present invention, component 140 may assess unrepresentativeness and self-inconsistency for resamples and selecting the best/ “most representative” resample.

[0029] In various embodiments of the present invention, downsampling component 140, via encoder 142, may build representations from genotype likelihoods (e.g., transforming genomic reads that are in FASTQ format into a numerical vector representation). Downsampling component 140, via analytics engine 146, may integrate mapping positional information and genotype likelihoods to identify the optimum vector representation of a resample. In various embodiments of the present invention, downsampling component 140, via decoder 144, may retrieve and/or receive as an input the most representative sample and produces a reconstruction of the original input reads. In various embodiments of the present invention, downsampling component 140, via decoder 144, may decode the vector representation of the optimum resample to obtain a down-sampled read file that resembles and maintains the genomic integrity of the original file.

[0030] In various embodiments of the present invention, given several reads overlapping a genomic location, the algorithm estimates genotype likelihood probabilities per read by maximizing the joint likelihood function parameters. These parameters are based on the Poisson

Graphical Distribution (PGD). The likelihood of the parameters  $\theta_0, \dots, \theta_{D-1}$  is the probability that the read was sampled from a distribution with these particular parameters. This can be simplified in Equation 1 as follows:

Equation 1:

$$L(\theta_1 \dots \theta_n) = \prod_{i=1}^N \frac{e^{-\theta_{1n_1}(\theta_1 n_1)k}}{k!} \prod_{i=1}^N \frac{e^{-\theta_{2n_2}(\theta_2 n_2)k}}{k!} \prod_{i=1}^N \frac{e^{-\theta_{3n_3}(\theta_3 n_3)k}}{k!} \dots \dots \dots \prod_{i=1}^N \frac{e^{-\theta_{nn_n}(\theta_n n_n)k}}{k!}$$

The test statistic may stay the same if the likelihood values are computed by pooling together genotype counts across multiple reads. For each read, a probability is estimated for each possible likelihood genotype.

**[0031]** In various embodiments of the present invention, resamples are generated using bootstrapping methods. For each bootstrapping iteration, given a FASTQ  $X_N$  where X is the FASTQ file with N total number of reads, a portion of reads is randomly selected to produce  $x_{1n}$ , sampling with replacement (FIG. 2B). This process is repeated I number for times. In the next iteration of the resampling procedure  $x_n$  is selected by increasing the number of reads.

**[0032]** To find the best  $Z_i$  (i.e., Z where Z is the best set of resamples obtained from the bootstrapping), self-inconsistency and unrepresentativeness are applied where,

Self-inconsistency is defined using Equation 2:

Equation 2:

$$\mathcal{D}_{si}(N, K, N_r) = \frac{1}{N_r(N_r - 1)} \sum_{p \neq q \forall p, q \in \{1, 2, \dots, N_r\}} (D_{LK}(\theta_k(x_{ip}), \theta_k(x_{iq})))$$

The average over self-inconsistency is then taken using Equation 3:

Equation 3:

$$\mathcal{D}_s(N, K) = \frac{1}{m} \sum_i \mathcal{D}_{si}(N, k, N_r)$$

Unrepresentativeness is defined using Equation 4:

Equation 4:

$$\mathcal{D}_{ri}(N, K, N_r) = \frac{1}{N_r} \sum_{\forall p, q \{1, 2, \dots, N_r\}} (D_{LK}(\theta_k(X_i), \theta_k(X_i)))$$

The average over self-inconsistency is then taken using Equation 5:

Equation 5:

$$\mathcal{D}_r(N, K) = \frac{1}{m} \sum_i \mathcal{D}_{ri}(N, k, N_r)$$

**[0033]** It should be noted that in Equation 1  $N$  represents the sample size ( the number of reads in the original file),  $n$  represents the resample size ( the number of reads in the resample),  $\theta$  represents the Poisson rate. It should be noted that in Equations 2-5  $N_r$  represents the number of resamples,  $N$  represents the number of sequences/reads in each,  $K$  represents the k-mer size,  $M$  represents the total number of samples,  $\mathcal{D}_{ri}$  represents unrepresentativeness , and  $\mathcal{D}_s$  represents self-inconsistency. In various embodiments of the present invention, a set of resamples are selected to be further processed by analytics engine 146. In various embodiments of the present invention, the input into analytics engine 146 is a set of resamples (e.g. 100 resamples each containing 1 million reads (that is a selected set of 100 resamples, and each resample contains 1 million reads)). Analytics engine 146 may then process the 100 resamples to get the best resample that is truly representative of the original file. This is done by minimizing the steps in the equations as described below.

**[0034]** To get the best resample or set of resamples (i.e., optimal resample(s)), wherein the best resample or set of resamples are resample(s) with the lowest values of unrepresentativeness and self-inconsistency, positional information and genotype likelihoods are combined and analyzed as follows:

**[0035]** Given  $Q_\theta(Z|X)$  where  $Z$  is the best set of resamples obtained from the bootstrapping and  $z_1, \dots, z_n$  are single resamples with  $m$  reads and  $X$  is the whole FASTQ file and  $\mathcal{P}_\theta(X|[z, y])$ , where  $y$  is the constraint obtained from alignment, which includes positional information of reads and their corresponding genotypic information minimize:  $\mathcal{D}_{KL}[q_\theta([z, y]|X) \mathcal{P}_\theta(X|[z, y])]$  which implies  $[z, y] \sim X$ , such that  $q_\theta([z, y]|X)$  at its minimum gives the best  $[z, y]$  considering one or more constraints. In various embodiments of the present invention, the best  $[z, y]$  is passed through decoder 144, which produces a down-sampled FASTQ file with a

reduced number of reads, but the reads that are retained maintain the genomic integrity of the input file.

**[0036]** In other embodiments of the present invention, downsampling component 140 may further use one or more constraints to identify an optimal set of resamples (i.e., best Z). The one or more constraints that may be utilized by downsampling component 140 to identify an optimal set of resamples may comprise: (i) selecting the largest interval based on the positional information encoded from the mapping; (ii) calculating the distribution of the intervals in each resample such that positional coverage is maximized; (iii) finding a resample with minimum overlap that has the biggest coverage distribution; (iv) calculating the average read weight, based on the genotype likelihoods occurring on each read; and/or calculating the overall weight of the resample by considering the weights of the reads.

**[0037]** FIG. 2A illustrates a block diagram depicting downsampling component 140 in communication with computing device 110 and/or server computer 120, within distributed data processing environment 100 of FIG. 1, for downsampling DNA sequence data using variational autoencoders. FIG. 2A provides one specific example of an overview of the steps followed to down-sample a FASTQ file using modified variational autoencoders. FIG. 2A provides an illustration of one implementation and does not imply any limitations with regard to the environments in which different embodiments may be implemented. Many modifications to the depicted environment may be made by those skilled in the art without departing from the scope of the invention as recited by the claims.

**[0038]** In step 201, downsampling component 140 receives and/or retrieves DNA sequence data. In various embodiments of the present invention, downsampling component 140 may receive and/or retrieve DNA sequence data from a database or as input from a user or another program.

**[0039]** In step 202, downsampling component 140 performs bootstrapping on the DNA sequence data. In various embodiments of the present invention, downsampling component 140, via encoder 142, performs bootstrapping on the received and/or retrieved DNA sequence data. In various embodiments of the present invention, downsampling component 140 uses a modified variational encoder (e.g., encoder 142) to perform probabilistic resampling to obtain a set of

resamples containing a plurality of reads, depicted in FIG. 2B. The selected resamples may be self-consistent and representative of the original file, wherein self-consistency describes the minimum variation occurring among resamples of the same size for a given number of resampling iterations and representativeness is the minimization of resample variation with respect to the original file. In various embodiments of the present invention, downsampling component 140 ensures that its genomic integrity is maintained using Equations 2-5 (described above). To obtain a set of resamples that are self-consistent, downsampling component 140 compares the similarity in distribution of the samples, and in doing so, it uses hash tables in the encoder to perform the distributional computation.

[0040] For example, FIG. 2B, provides an example of the overview of the bootstrapping procedure. In FIG. 2B, sample 1, represented as  $\theta_k(X_1)$ , undergoes random sampling 250 and produces an amount 'n' of bootstrap sample, where 'n' is a predetermined amount. In this particular example, encoder 142 outputs bootstrap sample 251 represented as  $\theta_k(X_{11})$ , bootstrap sample 252 represented as  $\theta_k(X_{12})$ , and bootstrap sample 253 represented as  $\theta_k(X_{1NR})$ .

[0041] In step 204, downsampling component 140 generates biological information. In various embodiments of the present invention, downsampling component 140 generates biological information further described in FIG. 2C. Biological information may include positional information, genotype likelihood, and/or any other biological information known in the art.

[0042] In step 206, downsampling component 140 generates files with positional and genotype information. In various embodiments of the present invention, downsampling component 140 generates files with positional and genotype information further described in FIG. 2C.

[0043] In step 208, downsampling component 140 integrates mapping positional information and genotype likelihoods. In various embodiments, downsampling component 140, via analytics engine 146, may integrate mapping positional information and genotype likelihoods to identify the optimum vector representation of a resample. Analytics engine 146 calculates interval distribution for a plurality of reads in each resample such that nucleotide positional coverage is maximized and finds a plurality of reads in a resample with minimum overlap that has the

biggest nucleotide coverage distribution for a given sample genome. Analytics engine 146 may also calculate the average read weight, based on a plurality of genotype likelihoods occurring on each read and overall weight of the resample based on the weights of a plurality of reads.

**[0044]** In step 210, downsampling component 140 can identify the best Z. In various embodiments of the present invention, downsampling component 140, via analytics engine 146, may identify the best Z. In various embodiments, downsampling component 140, via analytics engine 146, may calculate and display a list of the best Z's to a user.

**[0045]** In step 212, downsampling component 140 decodes the vector representation of an optimum resample. In various embodiments of the present invention, downsampling component 140, via decoder 144, may decode the vector representation of the optimum resample to obtain a down-sampled read file that resembles and maintains the genomic integrity of the original file.

**[0046]** FIG. 2C illustrates a block diagram depicting downsampling component 140 in communication with computing device 110 and/or server computer 120, within distributed data processing environment 100 of FIG. 1, for downsampling DNA sequence data using variational autoencoders. FIG. 2C provides an overview of biological information generation, wherein read fragments are mapped and the read fragment's positional information and genotype likelihoods are encoded into a numerical representation. FIG. 2C provides an illustration of one implementation and does not imply any limitations with regard to the environments in which different embodiments may be implemented. Many modifications to the depicted environment may be made by those skilled in the art without departing from the scope of the invention as recited by the claims.

**[0047]** In step 204, downsampling component 140 generates biological information. In various embodiments of the present invention, downsampling component 140 enables resampling DNA fragments generated from massively parallel sequencing, without compromising the genomic integrity of the original file.

**[0048]** In step 204A, downsampling component 140 aligns nucleotides. In various embodiments of the present invention, downsampling component 140 utilizes a modified variational encoder (e.g., encoder 142), where in encoder 142 aligns a plurality of reads containing nucleotides to different locations on a sample genome.

**[0049]** In step 204B downsampling component 140 generates positional information for the aligned plurality of reads. In various embodiments of the present invention, downsampling component 140, via encoder 142, may generate and/or identify positional information for the nucleotides in the aligned plurality of reads, wherein positional information comprises: mapping positions, mapping scores and nucleotide quality scores of a plurality of reads that map. In various embodiments of the present invention, downsampling component 140, via encoder 142, may encode mapping positions, mapping scores and nucleotide quality scores of a plurality of reads that map to a specific location of a sample genome into a vector representation. For example, a FASTQ file with millions of reads is mapped to a reference genome using the Smith-Waterman algorithm, as it is known in the art. The Smith-Waterman algorithm performs local sequence alignment to determine similar regions between two strings of nucleic acid sequences. The mapping position of each read is then encoded into a matrix as a start and end position (matrix 220). In matrix 220, the matrix structure comprises start and end positions of reads with data structure [n1, and n2] intervals continuously until the biological data and/or plurality of reads are represented, wherein n may be any positional, integer and/or numerical representation known in the art.

**[0050]** In step 204C, downsampling component 140 obtains genotype likelihoods for one or more genomic locations from the plurality of reads, wherein genotype likelihoods are a plurality of alleles with specified positional change on the plurality of reads. In various embodiments of the present invention, encoder 142 is modified to use the Poisson Graphical Distribution model (Equation 1) to obtain genotype likelihoods for each genomic location from a plurality of reads. In various embodiments of the present invention, downsampling component 140, via encoder 142, determines that a genotype likelihood matches a plurality of reads, where a genotype likelihood gives a probability of an allele occurring at a nucleotide position and discards low genotype likelihood probabilities that are not supported by a plurality of sequence reads.

**[0051]** In various embodiments of the present invention, a modified encoder (e.g., encoder 142) uses a plurality of genotype likelihoods on a read to calculate and determine read weight, down weighting reads with a few genotype likelihoods and discarding sequences without any genotype likelihoods estimated at any nucleotide position and sequences that do not map to any location on the sample genome to minimize redundancy. In various embodiments of the present

invention, a modified encoder (e.g., encoder 142) may be used to build representations from genotype likelihoods. The representations capture the genomic nuances in the original file such that the genomic variation in the original file is maintained in the down-sampled representations.

**[0052]** To calculate genotype likelihoods on each read, positions of the reference genome are not considered independently, but rather in consort to ensure that accumulation of genotypes is taken into consideration at the read level. This enables positional read anomalies to be viewed as dependent instead of assuming independency. Assuming that the occurrences of reads overlapping positions where there is an accumulation of genotypes is random, the genotype likelihoods can be modelled using PGD. A PGD pulls together the count of reads overlapping a position taking into consideration their dependence.

**[0053]** In addition, PGDs treat reads overlapping a genomic position in relation to the reference as an exponential family of multinomial joint distributions. This enables modelling the dependencies between the count of reads overlapping a location assuming an upper bound, which is reasonable to make especially considering read depth. A joint likelihood function may estimate the likelihood genotype call as a probability of a change from the reference to the alternate allele. Reads overlapping a genotype location are encoded into a matrix (e.g., matrix 222) that is used to train a model for estimating genotype likelihood probabilities. The estimated genotype likelihood probabilities are also used for weighting reads. Thus, the majority of reads that have accumulated more genotype likelihoods ideally can be found in the same re-sample, ensuring that the genomic integrity of the original file is maintained. Resamples with higher than average accumulated genotype likelihoods are weighted upwards to increase their chance of selection.

**[0054]** FIG. 3 illustrates operational steps of downsampling component 140, generally designated 300, in communication with computing device 110, within distributed data processing environment 100 of FIG. 1, for automatically downsampling DNA sequence data using variational autoencoders, in accordance with an embodiment of the present invention. FIG. 3 provides an illustration of one implementation and does not imply any limitations with regard to the environments in which different embodiments may be implemented. Many modifications to the depicted environment may be made by those skilled in the art without departing from the scope of the invention as recited by the claims.

[0055] In step 302, downsampling component 140 retrieves genomic sequence data from a file. In various embodiments of the present invention, downsampling component 140 may retrieve and/or receive genomic sequence data from one or more files, one or more databases, one or more search engines, and/or manually input data.

[0056] In step 304, downsampling component 140 executes bootstrapping to produce resamples. In various embodiments of the present invention, downsampling component 140, via encoder 142, may perform bootstrapping on the received and/or retrieved genomic sequence data to produce resamples. In various embodiments of the present invention, downsampling component 140, via a modified variational encoder (e.g., encoder 142) may perform probabilistic resampling to obtain a set of resamples containing a plurality of reads.

[0057] In step 306, downsampling component 140 resamples DNA fragments. In various embodiments of the present invention, downsampling component 140 may resample DNA fragments generated from parallel sequencing without compromising the genomic integrity of the original file.

[0058] In step 308, downsampling component 140 generates biological information. In various embodiments of the present invention, downsampling component 140, via encoder 142, may generate biological information wherein read fragments are mapped and the read fragment's positional information and genotype likelihoods are encoded into a numerical representation, as shown in FIG. 2C.

[0059] In step 310, downsampling component 140 generates a file containing positional and genotype information. In various embodiments of the present invention, downsampling component 140, via encoder 142, may generate one or more files containing positional and genotype information.

[0060] In step 312, downsampling component 140 accesses unrepresentativeness and self-inconsistency of the produced resamples. In various embodiments of the present invention, downsampling component 140, via encoder 142, may access the unrepresentativeness and self-inconsistency of one or more of the produced resamples and selecting one or more representative resample.

[0061] In step 314, downsampling component 140 builds representation from genotype likelihoods. In various embodiments of the present invention, downsampling component 140, via

encoder 142 (e.g., a modified encoder), may build one or more vector representations from the genotype likelihoods.

**[0062]** In step 316, downsampling component 140 integrates mapping positional information and genotype likelihoods. In various embodiments of the present invention, downsampling component 140, via analytics engine 146, may integrate mapping positional information and genotype likelihoods to identify one or more optimum vector representations of one or more resamples.

**[0063]** In step 318, downsampling component 140 decodes vector representation of resamples. In various embodiments of the present invention, downsampling component 140, via decoder 144, may receive as input the representative sample and produce a reconstruction of the original input reads. In various embodiments of the present invention, downsampling component 140, via decoder 144 (e.g., a modified decoder), decode vector representation of the optimum resample to obtain a down-sampled read file that resembles and maintains the genomic integrity of the original file.

**[0064]** In step 320, downsampling component 140 outputs a down-sampled read file. In various embodiments of the present invention, downsampling component 140, via decoder 144, may output one or more down sample read files, that resembles and maintains the genomic integrity of the original file, to one or more users by displaying the data on UI 106 and/or displaying response prompts alerting the user of the file data. In various embodiments of the present invention, downsampling component 140, via decoder 144, may execute the one or more down sample read files, that resembles and maintains the genomic integrity of the original file.

**[0065]** FIG. 4 depicts a block diagram of components of server computer 120 within distributed data processing environment 100 of FIG. 1, in accordance with an embodiment of the present invention. It should be appreciated that FIG. 4 provides only an illustration of one implementation and does not imply any limitations with regard to the environments in which different embodiments may be implemented. Many modifications to the depicted environment may be made.

**[0066]** FIG. 4 depicts computer system 400, where server computing 120 represents an example of computer system 400 that includes downsampling component 140. The computer system includes processors 401, cache 403, memory 402, persistent storage 405,

communications unit 407, input/output (I/O) interface(s) 406, display 409, external device(s) 408 and communications fabric 404. Communications fabric 404 provides communications between cache 403, memory 402, persistent storage 405, communications unit 407, and input/output (I/O) interface(s) 406. Communications fabric 404 may be implemented with any architecture designed for passing data and/or control information between processors (such as microprocessors, communications, and network processors, etc.), system memory, peripheral devices, and any other hardware components within a system. For example, communications fabric 404 may be implemented with one or more buses or a crossbar switch.

[0067] Memory 402 and persistent storage 405 are computer readable storage media. In this embodiment, memory 402 includes random access memory (RAM). In general, memory 402 may include any suitable volatile or non-volatile computer readable storage media. Cache 403 is a fast memory that enhances the performance of processors 401 by holding recently accessed data, and data near recently accessed data, from memory 402.

[0068] Program instructions and data used to practice embodiments of the present invention may be stored in persistent storage 405 and in memory 402 for execution by one or more of the respective processors 401 via cache 403. In an embodiment, persistent storage 405 includes a magnetic hard disk drive. Alternatively, or in addition to a magnetic hard disk drive, persistent storage 405 may include a solid-state hard drive, a semiconductor storage device, read-only memory (ROM), erasable programmable read-only memory (EPROM), flash memory, or any other computer readable storage media that is capable of storing program instructions or digital information.

[0069] The media used by persistent storage 405 may also be removable. For example, a removable hard drive may be used for persistent storage 405. Other examples include optical and magnetic disks, thumb drives, and smart cards that are inserted into a drive for transfer onto another computer readable storage medium that is also part of persistent storage 405.

[0070] Communications unit 407, in these examples, provides for communications with other data processing systems or devices. In these examples, communications unit 407 includes one or more network interface cards. Communications unit 407 may provide communications through the use of either or both physical and wireless communications links. Program instructions and

data used to practice embodiments of the present invention may be downloaded to persistent storage 405 through communications unit 407.

[0071] I/O interface(s) 406 enables for input and output of data with other devices that may be connected to each computer system. For example, I/O interface 406 may provide a connection to external devices 408 such as a keyboard, keypad, a touch screen, and/or some other suitable input device. External devices 408 may also include portable computer readable storage media such as, for example, thumb drives, portable optical or magnetic disks, and memory cards. Software and data used to practice embodiments of the present invention may be stored on such portable computer readable storage media and may be loaded onto persistent storage 405 via I/O interface(s) 406. I/O interface(s) 406 also connect to display 409.

[0072] Display 409 provides a mechanism to display data to a user and may be, for example, a computer monitor.

[0073] The programs described herein are identified based upon the application for which they are implemented in a specific embodiment of the invention. However, it should be appreciated that any particular program nomenclature herein is used merely for convenience, and thus the invention should not be limited to use solely in any specific application identified and/or implied by such nomenclature.

[0074] The present invention may be a system, a method, and/or a computer program product. The computer program product may include a computer readable storage medium (or media) having computer readable program instructions thereon for causing a processor to carry out aspects of the present invention.

[0075] The computer readable storage medium may be any tangible device that may retain and store instructions for use by an instruction execution device. The computer readable storage medium may be, for example, but is not limited to, an electronic storage device, a magnetic storage device, an optical storage device, an electromagnetic storage device, a semiconductor storage device, or any suitable combination of the foregoing. A non-exhaustive list of more specific examples of the computer readable storage medium includes the following: a portable computer diskette, a hard disk, a random access memory (RAM), a read-only memory (ROM), an erasable programmable read-only memory (EPROM or Flash memory), a static random

access memory (SRAM), a portable compact disc read-only memory (CD-ROM), a digital versatile disk (DVD), a memory stick, a floppy disk, a mechanically encoded device such as punch-cards or raised structures in a groove having instructions recorded thereon, and any suitable combination of the foregoing. A computer readable storage medium, as used herein, is not to be construed as being transitory signals *per se*, such as radio waves or other freely propagating electromagnetic waves, electromagnetic waves propagating through a waveguide or other transmission media (e.g., light pulses passing through a fiber-optic cable), or electrical signals transmitted through a wire.

[0076] Computer readable program instructions described herein may be downloaded to respective computing/processing devices from a computer readable storage medium or to an external computer or external storage device via a network, for example, the Internet, a local area network, a wide area network and/or a wireless network. The network may comprise copper transmission cables, optical transmission fibers, wireless transmission, routers, firewalls, switches, gateway computers and/or edge servers. A network adapter card or network interface in each computing/processing device receives computer readable program instructions from the network and forwards the computer readable program instructions for storage in a computer readable storage medium within the respective computing/processing device.

[0077] Computer readable program instructions for carrying out operations of the present invention may be assembler instructions, instruction-set-architecture (ISA) instructions, machine instructions, machine dependent instructions, microcode, firmware instructions, state-setting data, or either source code or object code written in any combination of one or more programming languages, including an object oriented programming language such as Smalltalk, C++ or the like, and conventional procedural programming languages, such as the "C" programming language or similar programming languages. The computer readable program instructions may execute entirely on the user's computer, partly on the user's computer, as a stand-alone software package, partly on the user's computer and partly on a remote computer or entirely on the remote computer or server. In the latter scenario, the remote computer may be connected to the user's computer through any type of network, including a local area network (LAN) or a wide area network (WAN), or the connection may be made to an external computer (for example, through the Internet using an Internet Service Provider). In some embodiments,

electronic circuitry including, for example, programmable logic circuitry, field-programmable gate arrays (FPGA), or programmable logic arrays (PLA) may execute the computer readable program instructions by utilizing state information of the computer readable program instructions to personalize the electronic circuitry, in order to perform aspects of the present invention.

**[0078]** Aspects of the present invention are described herein with reference to flowchart illustrations and/or block diagrams of methods, apparatus (systems), and computer program products according to embodiments of the invention. It will be understood that each block of the flowchart illustrations and/or block diagrams, and combinations of blocks in the flowchart illustrations and/or block diagrams, may be implemented by computer readable program instructions.

**[0079]** These computer readable program instructions may be provided to a processor of a general purpose computer, a special purpose computer, or other programmable data processing apparatus to produce a machine, such that the instructions, which execute via the processor of the computer or other programmable data processing apparatus, create means for implementing the functions/acts specified in the flowchart and/or block diagram block or blocks. These computer readable program instructions may also be stored in a computer readable storage medium that may direct a computer, a programmable data processing apparatus, and/or other devices to function in a particular manner, such that the computer readable storage medium having instructions stored therein comprises an article of manufacture including instructions which implement aspects of the function/act specified in the flowchart and/or block diagram block or blocks.

**[0080]** The computer readable program instructions may also be loaded onto a computer, other programmable data processing apparatus, or other device to cause a series of operational steps to be performed on the computer, other programmable apparatus or other device to produce a computer implemented process, such that the instructions which execute on the computer, other programmable apparatus, or other device implement the functions/acts specified in the flowchart and/or block diagram block or blocks.

**[0081]** The flowchart and block diagrams in the FIG.s illustrate the architecture, functionality, and operation of possible implementations of systems, methods, and computer program products

according to various embodiments of the present invention. In this regard, each block in the flowchart or block diagrams may represent a module, a segment, or a portion of instructions, which comprises one or more executable instructions for implementing the specified logical function(s). In some alternative implementations, the functions noted in the blocks may occur out of the order noted in the FIG.s. For example, two blocks shown in succession may, in fact, be executed substantially concurrently, or the blocks may sometimes be executed in the reverse order, depending upon the functionality involved. It will also be noted that each block of the block diagrams and/or flowchart illustration, and combinations of blocks in the block diagrams and/or flowchart illustration, may be implemented by special purpose hardware-based systems that perform the specified functions or acts or carry out combinations of special purpose hardware and computer instructions.

**[0082]** The descriptions of the various embodiments of the present invention have been presented for purposes of illustration, but are not intended to be exhaustive or limited to the embodiments disclosed. Many modifications and variations will be apparent to those of ordinary skill in the art without departing from the scope and spirit of the invention. The terminology used herein was chosen to best explain the principles of the embodiment, the practical application or technical improvement over technologies found in the marketplace, or to enable others of ordinary skill in the art to understand the embodiments disclosed herein.

## CLAIMS

What is claimed is:

1. A computer-implemented method for automatically downsampling DNA sequence data using variational autoencoders and preserving genomic integrity of an original file, the method comprising:
  - executing, by an encoder, bootstrapping on genomic sequence data to produce resamples;
  - assessing, by the encoder, unrepresentativeness and self-inconsistency of the resamples and selecting a representative resample according to the assessment;
  - building, by a modified encoder, vector representations from genotype likelihoods based on the selected representative sample;
  - integrating, by an analytics engine, mapping positional information and the genotype likelihoods to identify an optimum vector representation of a resample; and
  - decoding, by a modified decoder, the identified optimum vector representation of the resample to obtain a down-sampled read file that resembles and maintains the genomic integrity of the original file.
2. The computer-implemented method of claim 1 further comprising:
  - resampling, by the encoder, DNA fragments generated from massively parallel sequencing, without compromising the genomic integrity of the original file.
3. The computer-implemented method of claim 1 further comprising:
  - executing, by a modified variational encoder, probabilistic resampling to obtain a set of resamples containing a plurality of reads.
4. The computer-implemented method of claim 1 further comprising:
  - receiving, by the decoder, the representative resample as input; and
  - producing, by the decoder, a reconstruction of a plurality of original input reads based on received representative resample.
5. The computer-implemented method of claim 1 further comprising:
  - outputting the down-sampled read file that resembles and maintains the genomic integrity of the original file to a user.

6. The computer-implemented method of claim 1 further comprising:  
generating, by the encoder, a file containing positional and genotype information.
7. The computer-implemented method of claim 1 further comprising;  
generating, by the encoder, biological information, wherein generating the biological information comprises:  
aligning, by a modified variational encoder, a plurality of reads containing nucleotides to different locations on a sample genome;  
generating, by the encoder, positional information for the nucleotides in the aligned plurality of reads, wherein the positional information comprises: mapping positions, mapping scores and nucleotide quality scores of a plurality of reads that map;  
encoding, by the encoder, the mapping positions, the mapping scores and the nucleotide quality scores of a plurality of reads that map to a specific location of a sample genome into a vector representation; and  
obtaining, by the encoder, genotype likelihoods for one or more genomic locations from the plurality of reads, wherein the genotype likelihoods are a plurality of alleles with specified positional change on the plurality of reads.
8. A computer program product for automatically downsampling DNA sequence data using variational autoencoders and preserving genomic integrity of an original file, the computer program product comprising:  
one or more computer readable storage devices and program instructions stored on the one or more computer readable storage devices, the stored program instructions comprising:  
program instructions to execute, by an encoder, bootstrapping on genomic sequence data to produce resamples;  
program instructions to assess, by the encoder, unrepresentativeness and self-inconsistency of the resamples and selecting a representative resample according to the assessment;  
program instructions to build, by a modified encoder, vector representations from genotype likelihoods based on the selected representative sample;

program instructions to integrate, by an analytics engine, mapping positional information and the genotype likelihoods to identify an optimum vector representation of a resample; and

program instructions to decode, by a modified decoder, the identified optimum vector representation of the resample to obtain a down-sampled read file that resembles and maintains the genomic integrity of the original file.

9. The computer program product of claim 8 further comprising:  
program instructions to resample, by the encoder, DNA fragments generated from massively parallel sequencing, without compromising the genomic integrity of the original file.
10. The computer program product of claim 8 further comprising:  
program instructions to execute, by a modified variational encoder, probabilistic resampling to obtain a set of resamples containing a plurality of reads.
11. The computer program product of claim 8 further comprising:  
program instructions to receive, by the decoder, the representative resample as input; and  
program instructions to produce, by the decoder, a reconstruction of a plurality of original input reads based on received representative resample.
12. The computer program product of claim 8 further comprising:  
program instructions to output the down-sampled read file that resembles and maintains the genomic integrity of the original file to a user.
13. The computer program product of claim 8 further comprising:  
program instructions to generate, by the encoder, a file containing positional and genotype information.
14. The computer program product of claim 8 further comprising:  
program instructions to generate, by the encoder, biological information, wherein generating the biological information comprises:

program instructions to align, by a modified variational encoder, a plurality of reads containing nucleotides to different locations on a sample genome;

program instructions to generate, by the encoder, positional information for the nucleotides in the aligned plurality of reads, wherein the positional information comprises: mapping positions, mapping scores and nucleotide quality scores of a plurality of reads that map;

program instructions to encode, by the encoder, the mapping positions, the mapping scores and the nucleotide quality scores of a plurality of reads that map to a specific location of a sample genome into a vector representation; and

program instructions to obtain, by the encoder, genotype likelihoods for one or more genomic locations from the plurality of reads, wherein the genotype likelihoods are a plurality of alleles with specified positional change on the plurality of reads.

15. A computer system for automatically downsampling DNA sequence data using variational autoencoders and preserving genomic integrity of an original file, the computer system comprising:

one or more computer processors;

one or more computer readable storage devices;

program instructions stored on the one or more computer readable storage devices for execution by at least one of the one or more computer processors, the stored program instructions comprising:

program instructions to execute, by an encoder, bootstrapping on genomic sequence data to produce resamples;

program instructions to assess, by the encoder, unrepresentativeness and self-inconsistency of the resamples and selecting a representative resample according to the assessment;

program instructions to build, by a modified encoder, vector representations from genotype likelihoods based on the selected representative sample;

program instructions to integrate, by an analytics engine, mapping positional information and the genotype likelihoods to identify an optimum vector representation of a resample; and

program instructions to decode, by a modified decoder, the identified optimum vector representation of the resample to obtain a down-sampled read file that resembles and maintains the genomic integrity of the original file.

16. The computer system of claim 15 further comprising:  
program instructions to resample, by the encoder, DNA fragments generated from massively parallel sequencing, without compromising the genomic integrity of the original file.
17. The computer system of claim 15 further comprising:  
program instructions to execute, by a modified variational encoder, probabilistic resampling to obtain a set of resamples containing a plurality of reads.
18. The computer system of claim 15 further comprising:  
program instructions to receive, by the decoder, the representative resample as input; and  
program instructions to produce, by the decoder, a reconstruction of a plurality of original input reads based on received representative resample.
19. The computer system of claim 15 further comprising:  
program instructions to output the down-sampled read file that resembles and maintains the genomic integrity of the original file to a user; and  
program instructions to generate, by the encoder, a file containing positional and genotype information.
20. The computer system of claim 15 further comprising;  
program instructions to generate, by the encoder, biological information, wherein generating the biological information comprises:  
program instructions to align, by a modified variational encoder, a plurality of reads containing nucleotides to different locations on a sample genome;  
program instructions to generate, by the encoder, positional information for the nucleotides in the aligned plurality of reads, wherein the positional information comprises: mapping positions, mapping scores and nucleotide quality scores of a plurality of reads that map;  
program instructions to encode, by the encoder, the mapping positions, the mapping scores and the nucleotide quality scores of a plurality of reads that map to a specific location of a sample genome into a vector representation; and

program instructions to obtain, by the encoder, genotype likelihoods for one or more genomic locations from the plurality of reads, wherein the genotype likelihoods are a plurality of alleles with specified positional change on the plurality of reads.