JARVIS2: a data compressor for large genome sequences

Diogo Pratas[1,2,3] and Armando J. Pinho[1,2]

[1]IEETA/LASI - Institute of Electronics and Informatics Engineering of Aveiro
[2]DETI - Department of Electronics, Telecommunications and Informatics
University of Aveiro, 3810-193 Aveiro, Portugal, {pratas,ap}@ua.pt
[3]DoV - Department of Virology, University of Helsinki, 00014 Helsinki, Finland

Abstract

Reducing data storage and its associated costs is one of the major bottlenecks in large-scale genomic projects. In this paper, we present a new method for efficient reference-free data compression of genomic sequences, using a mixture of multiple finite-context models (FCMs) and weighted local stochastic repeat models (WLSRMs). In the WLSRMs, we developed a new cache-hash memory model for increasing the compression ratio, while decreasing the RAM. The mixture is provided by a new architecture, that includes a neural network, followed by arithmetic encoding. The method is implemented using the C language as the JARVIS2 tool. Additionally, we provide a C/Bash implementation for using JARVIS2 in FASTA data. The results show an improvement in the compression of two extensive datasets, while using lower computational resources relative to the second best compression ratio tool. JARVIS2 is freely available at `https://github.com/cobilab/jarvis2`.

Introduction

Genomes are found in the most diverse places, for example, in extreme environments as uranium mines [1], in soft and hard tissues [2, 3], ancient cadavers [4], important food crops [5], marine environments [6], or deep subterranean habitats [7]. The environment and species interactions are key for genome adaptation, providing a wide diversity in characteristics, namely high copy number, high heterogeneity, high level of substitution mutations, or multiple rearrangements. Therefore, compressing genomic sequences requires the ability to model heterogeneous, dynamic, incomplete, and imperfect data [8], including the ability to model specific characteristics that can add substantial improvement upon general-purpose data compressors [9].

The genomic data compression field currently holds three decades of research; it started with Biocompress [10] in 1993. After Biocompress, several algorithms emerged, mostly modeling the existence of exact or approximate repeated and inverted repeated regions, namely with simple bit encoding, context modeling, or dictionary approaches. More than fifty genomic data compression methods have been described and some implemented into efficient tools [11–13]; for example, NMLComp [14], GeNML [15], DNA-COMPACT [16], XM [17], CoGi [18], the GeCo-series [19–21], DeepDNA [22], Jarvis [23].

With the increasing availability of multiple full genomes, the FASTA format emerged. This format permitted standardizing the co-existence of genomic sequences (in a visible horizontal range) along with annotations (headers). Usually, the biological sequence is by a large margin the most abundant part of these data and, hence, multiple tools use specialized compression algorithms combined with simple header coding; for example, Deliminate [24], MFCompress [25], NAF [26], and MBGC [27].

The importance of high-ratio and reference-free compression of genomic data goes beyond the usual purpose of reducing the data storage, also being crucial for analysis applications, such as genomic sequence classification [28], rearrangement detection [29], metagenomic inference [30], genome reconstruction [31], among others.

In this paper, we provide a data compression tool (JARVIS2) for genomic data, including an extension to handle FASTA data. We combine the sensitivity of specific models, namely the usage of context and repeat models, with the power of neural networks for context mixing. We apply caches to repeat models, being able to substantially decrease the computational memory, while providing competitive or improved results and additionally parallelize the methodology for reducing the computation time, both for compression and decompression.

Method

JARVIS2 uses a combination of multiple finite-context models (FCMs), including substitution tolerant context models (STCM) of several orders/depths, and weighted local stochastic repeat models (WLSRM). The combination of these models is provided by a neural network, that mixes the probabilities estimated by each model. The encoding is performed with an arithmetic encoder derived from [32]. The JARVIS2 decompression takes approximately the same computational time and memory as used by the JARVIS2 data compressor. JARVIS2 is made available for free download (GPv3 license) at `https://github.com/cobilab/JARVIS2`.

Figure 1 depicts the architecture of the method with identification of the layers for the memory, probabilistic, and weight models. Also, it includes the division according to the nature of the models, specifically FCMs and WLSRMs, which are explained in the following subsections.

Finite-context models (FCMs) and substitution tolerant context models (STCMs)

Consider a sequence, $x = x_1, \ldots, x_N$, over the alphabet $\Theta = \{\mathrm{A}, \mathrm{C}, \mathrm{G}, \mathrm{T}\}$. The goal is to minimize the data describing this sequence, without loss of information and without using external information, besides the decompression source code. To this end, combinations of FCMs and STCMs can be used to derive different probability estimates for observing a symbol x_i in a sequence, given an appropriate context.

A FCM is a probabilistic model relying on the Markov property, which provides the probability of the next symbol given a certain context depth. These models have been used extensively for encoding diverse types of data, including DNA sequences (e.g., [19, 25]). STCMs are probabilistic-algorithmic models that can be used along with FCMs when confronted with symbol substitutions in genomic sequences [19, 33]. These models can be automatically disabled, to reduce the number of calculations and to increase the performance of the proposed method. Such an operation is automatically performed using an array of size h (typically, the context size), which preserves the past h hits. Observing a symbol in the sequence, the memory is checked for the

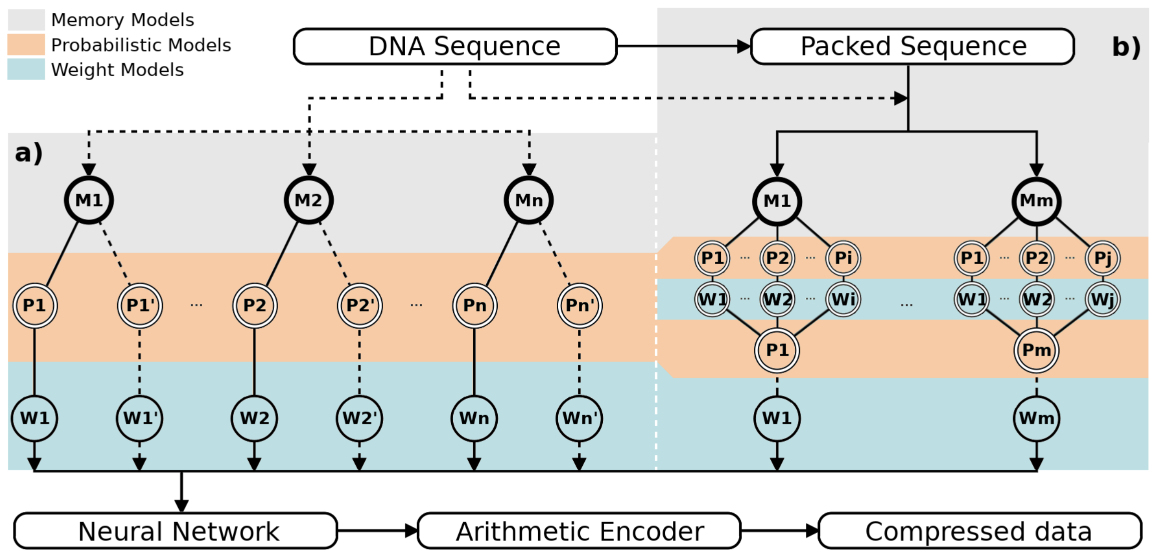

Figure 1: Architecture with the colored layers containing the Memory, Probabilistic, and Weight Models. a) the left panel represents the context and substitution tolerant context models (STCM); b) the right panel stands for the classes of local weighted stochastic repeat models (WLSRM). The packed sequence is the past sequence representation using two bits.

symbol with the highest number of occurrences. If they are equal, a hit is saved in the history array. Before getting to store a hit/miss in the array, it is checked for the number of misses and, in the case they are more than a predefined threshold t, the STCM will be disabled and the history array will be reset. This process is performed for each nucleotide in the sequence.

One of the advantages of using a combination of FCMs and STCMs is that, for the same context depth, the memory model can be shared, as shown in Figure 1.

Weighted local stochastic repeat models (WLSRMs)

The repeat model as been used in a few tools, namely XM [17], GReEn [34], and JARVIS [23]. This model stores into memory the positions relative to the sequence that has an identical k-mer identified in the past of the sequence. The positions are stored, based on a causal processing approach, usually using a hash-table. The model is used after a k-mer match occurs and is switched off after a certain threshold of performance is reached.

The repeat models are called stochastic because to start a new repeat (after a k-mer match), any position given the same k-mer has the same probability of being used. The stochastic nature enables uniform distribution of the repeats to start in different positions along the sequence. Another advantage is the absence of indexes to represent the position of the repeat being used under the positions vector. As such, the stochastic nature allows decreasing the memory inherent to the representation of the hash-table. Along with the hash-table of positions, the sequence needs to be continuously preserved in memory. To minimize its representation in memory, we pack each DNA symbol into two bits, instead of the common eight bits.

In this paper, we specifically developed local repeat stochastic models based on caches for JARVIS2. The cache enables to keep only the positions associated to the latest l past symbols. Therefore, the model forgets the k-mer entries that are being uploaded into the memory model when l is reached. Importantly, this approach is key for JARVIS2, since it enables to use smaller k-mers without loosing sensitivity, especially given the randomly selected position.

Finally, the repeat models are combined using a weighted approach. For each repeat, the weight is adapted according to its past performance under a decaying forgetting factor. The decaying forgetting factor is very small, since in this case the weights need to adapt fast.

Artificial Neural Network (ANN)

The JARVIS compressor [35] uses a competitive prediction context-based model to estimate, for each symbol, the best class of models to be used. Although the computation is fast, the accuracy is limited. In JARVIS2, we use an ANN that is imported from GeCo3 [21] with minor adaptations. Accordingly, we use a multilayer perceptron (MLP), which confers accurate and efficient predictions [36], while being one of the most resource-efficient neural networks regarding time and memory usage. Moreover, it is straightforward to implement and validate, and has proven to provide improved compression ratios both for genomic [21] and proteomic sequences [37].

Specifically, the network has a single hidden layer. The activation function is the sigmoid and the loss function is the mean squared error. The probabilities are stetched according to the works of Matt Mahoney (`http://mattmahoney.net/dc/dce.html`). The inputs to the network are the outputs of the FCMs, including the STCMs, and of the WLSRMs. One node per symbol is used as output from the network. After the result is transferred to the arithmetic encoder, the network is trained with the current symbol using the stochastic gradient descent algorithm without momentum [38]. This process requires only two parameters, namely the number of nodes of the hidden layer and the learning rate.

Extension for compressing FASTA data

FASTA data are composed of two channels of information: headers and genomic sequences. For compressing FASTA data, the JARVIS2 Bash extension splits and compresses them using different methods. The headers are compressed using the BBB (Big Block Burrows-Wheeler transform), developed by Matt Mahoney (`http://www.mattmahoney.net/dc/#bbb`), operating with a 40 MiB block size. Since genomic sequences may have different symbols than the DNA bases from Θ, an additional channel of information is created to represent extra symbols. When a symbol is a known DNA base, then, the extra channel uses the ASCII index of zero; otherwise, the self symbol is used (an extra symbol includes a newline). Usually, these symbols are rare and the majority of these sequences are constituted by zeros yielding a tiny amount of bits after compression employing Bzip2 (`https://sourceware.org/bzip2`). The genomic sequences are compressed with JARVIS2, using one of the available compression modes. All channels of information are compressed in parallel. Finally, the headers' compressed data, extra symbols, and genomic sequences are merged in a tar file. This permits to substitute JARVIS2 with another genomic compressor, enabling the use of genomic compressors as specialized FASTA compressors. The compression levels are the same used by the JARVIS2 binary. Additionally, the JARVIS2 FASTA extension can be parameterized according to a custom block size and number of threads.

Results

In this section, we evaluate the performance of JARVIS2 using two large benchmarks, namely the full human and cassava genomes. The complete benchmarks and replication scripts are publicly available at `https://github.com/cobilab/HumanGenome` and `https://github.com/cobilab/CassavaGenome`. These benchmarks contain the full tables, including computational time and memory. Both benchmarks have been performed with a desktop computer running Linux with Intel® Core™ i7-6700 CPU @ 3.40GHz × 8, 31.2 GiB RAM, and a disk of 3 TB. All the runs have been successfully decompressed without loss of information. In both benchmarks, we excluded computational tools that required more than 32 GB of RAM. This exclusion included several tools, such as Cmix and MBGC.

The first benchmark uses the human genome T2T sequence (Chm13 version 2.0) [39]. This sequence is composed of 3,117,292,120 DNA symbols from the alphabet $\{A, C, G, T\}$ corresponding to the concatenation of the whole human chromosomes. Assuming an uniform distribution, where all symbols are considered independent, we need $779,323,030$ bytes to represent the full sequence without loss of information. Therefore, this value serves as the baseline.

Although the sequence to compress contains approximately 779 MB of data (baseline), reducing this value without loss of information is not an easy task. This is proved by the inability of several tools to compress the data below this value. For example, Gzip uses more than 779 MB to compress the data. The reason is that these general-purpose data compression tools have models that are not designed to cope with the characteristics of the genomic sequences.

The selection of the main tools to use is based in benchmark reviews [11–13] and the compression challenge thread from the Data Compression community.

Figure 2 provides a selection of several benchmark entries, including the highest compression ratio for each tool. In this figure, we can notice that the general-purpose models bzip2 and lzma were not able to reduce the bps to less than 1.658 using the best compression parameters.

An interesting entry was made by bsc-m03 (`https://github.com/IlyaGrebnov/bsc-m03`) that showed reasonable compression capability in balance with the required computational resources. Two of the best high-ratio data compression tools, namely nncp and paq8l, achieved a bps of 1.514 and 1.571, respectively, while using much more computational time (at least one order of magnitude above).

The specific-purpose data compressors NAF (best compression mode), Jarvis and GeCo2 (optimized), and MFCompress (best compression mode) showed a minimum

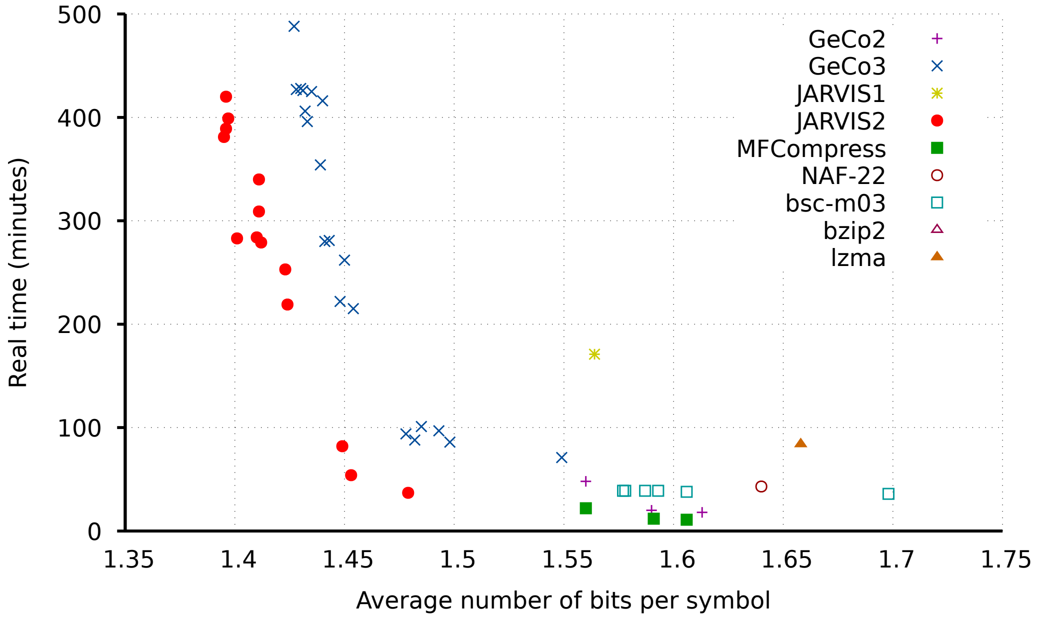

Figure 2: Human genome T2T compression benchmark (Time is for compression only).

bps of 1.560, led by MFCompress.

The first entry (Run1) of the GeCo3 tool provided a bps of 1.485, showing a clear advance over previous tools. The internal models of GeCo3 were parameterized based on our experience, namely with the use of two context models, with context orders of 3 and 19, and a tolerant context model with a context order of 19. The models were blended with a neural network and the cache-hash was set to a maximum of 20 collisions. To improve the GeCo3 compression factor, we optimized the models of GeCo3, namely increasing the cache-hash at the expense of higher computational time and RAM. We also added more context models while combining different orders and tuned parameters used in the probability estimation. Finally, after increasing the neural network and tuning the learning rate, we achieved a bps of 1.425 (Run39).

Although we could reduce more than 0.06 bps using substantially less computational time than the second best tool, we were aware of reaching close to the best that these models could provide while spending considerable time and RAM.

The first entry of JARVIS2 (without many optimizations), reached a lower bps than the best GeCo3, while using less computational time. After parameter optimization, we reached a version (Run47) with a bps of 1.410, while using less than half of the computational time and memory used in the best GeCo3 entry. Another entry (Run46) provided a lower bps than the best GeCo3 version using one-third of the time and RAM.

Finally, we optimized the models for an iterative search of the highest compression and reached the best bps of 1.395 (Run52). As far as we know, this is the best value achieved so far.

The second benchmark (Figure 3) is performed in the recent haplotype-resolved sequenced Cassava (TME 204) complete genome [5]. This genome comprises over 762,392,783 DNA symbols. Assuming an uniform distribution, where all symbols are considered independent, we need 95, 299, 097 bytes to represent the full sequence without loss of information. Therefore, this value serves as the baseline.

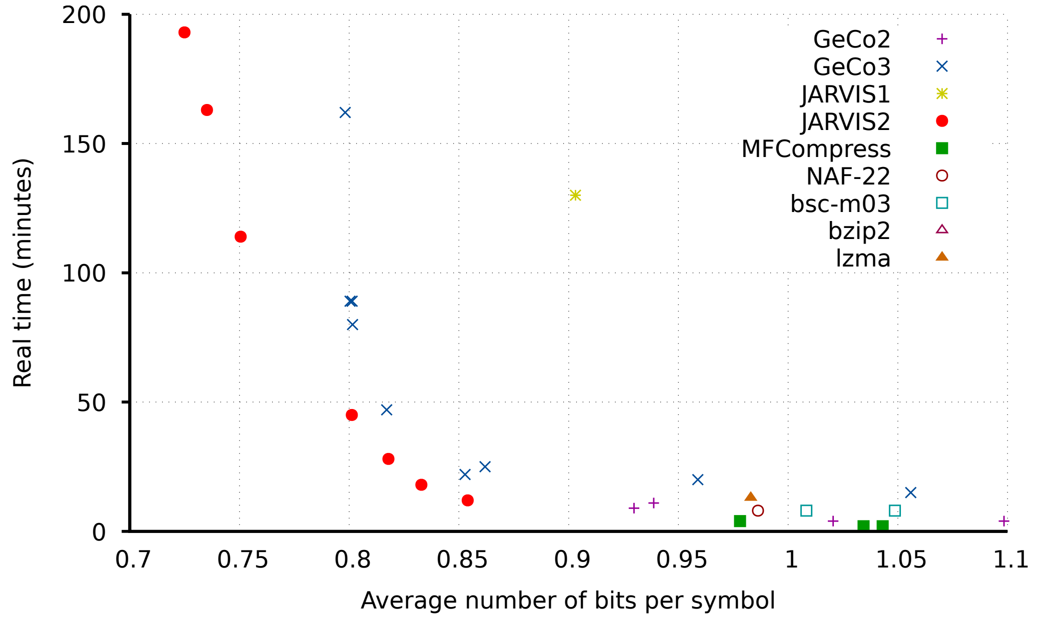

Figure 3: Cassava genome TME204 compression benchmark (Time is for compression only).

For running the JARVIS2 benchmark, we selected some of the best runs from the human benchmark. Despite this selection, the nature of the cassava genome is different, being characterized by extensive rearrangements and higher redundancy.

Generally, all the data compression tools behaved similar to the human benchmark, with a substantial reduction of the bps, roughly, half.

Similar to the human benchmark, in the cassava benchmark, JARVIS2 reached the lowest bps. The best JARVIS2 entry was able to achive a bps of 0.725 (Run18), while the second best tool, GeCo3, in the optimized version, was able to attain 0.798 (Run7), although using extensively more RAM.

Although not presented in this article, the decompression speed can be a very important feature, for example, for the integration of decompression in data analysis pipelines. GeCo2, GeCo3, JARVIS1, JARVIS2, and MFCompress are symmetric, using approximately the same time and RAM for compression and decompression. However, many other compressors are not symmetric, and some focus on providing extremely fast decompression (for example, NAF and lzma).

Therefore, the utility of JARVIS2 is currently at the medium- and long-term storage of the data, specifically of individual genomes, while for collections of similar genomes MBGC and NAF are good alternatives. Since JARVIS2 predicts probabilities for each symbol to compress with the highest average compression ratio according to the tests we have made, it can be used to predict, compare, relate, and locate motifs with higher accuracy in important computational biology applications.

Conclusions

In this paper, we presented JARVIS2, a reference-free genomic sequence compression tool with an extension to compress losslessly data in FASTA format. We described the methodology of JARIVS2, characterized by compression using multiple finite-context models (FCMs) and weighted local stochastic repeat models (WLSRMs). These models are mixed using a custom Artificial Neural Network (ANN) followed by arithmetic encoding. Specifically, in the WLSRMs, we developed a new memory model based on a cache-hash that efficiently decreased the RAM associated with previous implementations and increased the compression ratio. We showed an increased compression of all the datasets while reducing computational resources. Moreover, we showed that JARVIS2 presents the best-known data compression ratios on the complete human and cassava genomes.

Acknowledgements

The authors wish to thank the Finnish Computing Competence Infrastructure (FCCI) for supporting this project with computational and data storage resources. D.P. is funded by national funds through FCT – Fundação para a Ciência e a Tecnologia, I.P., under the Scientific Employment Stimulus - Institutional Call - reference CEECINST/00026/2018. We also acknowledge national funds through the FCT in the context of the project UIDB/00127/2020.

References

[1] C Covas, T Caetano, A Cruz, T Santos, L Dias, et al., "Pedobacter lusitanus sp. nov., isolated from sludge of a deactivated uranium mine," *International Journal of Systematic and Evolutionary Microbiology*, vol. 67, no. 5, pp. 1339–1348, 2017.

[2] L Pyöriä, M Jokinen, M Toppinen, H Salminen, T Vuorinen, et al., "HERQ-9 Is a New Multiplex PCR for Differentiation and Quantification of All Nine Human Herpesviruses," *mSphere*, vol. 5, no. 3, 2020.

[3] M Toppinen, D Pratas, E Väisänen, M Söderlund-Venermo, K Hedman, M F Perdomo, and A Sajantila, "The landscape of persistent human DNA viruses in femoral bone," *Forensic Science International: Genetics*, p. 102353, 2020.

[4] A T Duggan, M F Perdomo, D Piombino-Mascali, S Marciniak, D Poinar, et al., "17th century variola virus reveals the recent history of smallpox," *Current Biology*, vol. 26, no. 24, pp. 3407–3412, 2016.

[5] W Qi, Y Lim, A Patrignani, P Schläpfer, A Bratus-Neuenschwander, S Grüter, et al., "The haplotype-resolved chromosome pairs of a heterozygous diploid African cassava cultivar reveal novel pan-genome and allele-specific transcriptome features," *GigaScience*, vol. 11, 2022.

[6] H Teixeira, T Berg, L Uusitalo, K Fürhaupter, A Heiskanen, et al., "A catalogue of marine biodiversity indicators," *Frontiers in Marine Science*, vol. 3, pp. 207, 2016.

[7] D A Cowan, J Ramond, T P Makhalanyane, and P De Maayer, "Metagenomics of extreme environments," *Current Opinion in Microbiology*, vol. 25, pp. 97–102, 2015.

[8] A Golan, *Foundations of Info-Metrics: Modeling and Inference with Imperfect Information*, Oxford University Press, 2017.

[9] Goyal M, Tatwawadi K, Chandak S, and Ochoa I, "Deepzip: Lossless data compression using recurrent neural networks," in *Data Compression Conference (DCC'19). IEEE,*, Ali Bilgin, James A. Storer, Michael W. Marcellin, and Joan Serra-Sagrista, Eds., United States, 5 2019, Data Compression Conference Proceedings, Institute of Electrical and Electronics Engineers Inc.

[10] S Grumbach and F Tahi, "Compression of DNA sequences," in *Data Compression Conference (DCC'93). IEEE,*, 1993, pp. 340–350.

[11] K Kryukov, M T Ueda, S Nakagawa, and T Imanishi, "Sequence Compression Benchmark (SCB) database—A comprehensive evaluation of reference-free compressors for FASTA-formatted sequences," *GigaScience*, vol. 9, no. 7, pp. giaa072, 2020.

[12] M Hernaez, D Pavlichin, T Weissman, and I Ochoa, "Genomic data compression," *Annual Review of Biomedical Data Science*, vol. 2, pp. 19–37, 2019.

[13] M Hosseini, D Pratas, and A J Pinho, "A survey on data compression methods for biological sequences," *Information*, vol. 7, no. 4, pp. 56, 2016.

[14] I Tabus, G Korodi, and J Rissanen, "DNA sequence compression using the normalized maximum likelihood model for discrete regression," in *Data Compression Conference (DCC'03). IEEE,*, 2003, pp. 253–262.

[15] G Korodi and I Tabus, "Normalized maximum likelihood model of order-1 for the compression of DNA sequences," in *Data Compression Conference (DCC'07). IEEE,*, Mar. 2007, pp. 33–42.

[16] P Li, S Wang, J Kim, H Xiong, L Ohno-Machado, and X Jiang, "DNA-COMPACT: DNA Compression Based on a Pattern-Aware Contextual Modeling Technique," *PloS One*, vol. 8, no. 11, pp. e80377, 2013.

[17] M. D. Cao, T. I. Dix, L. Allison, and C. Mears, "A simple statistical algorithm for biological sequence compression," in *Data Compression Conference (DCC'07). IEEE,*, Mar. 2007, pp. 43–52.

[18] X Xie, S Zhou, and J Guan, "CoGI: Towards compressing genomes as an image," *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, vol. 12, no. 6, pp. 1275–1285, 2015.

[19] D Pratas, A J Pinho, and P J S G Ferreira, "Efficient compression of genomic sequences," in *Data Compression Conference (DCC'16). IEEE,*.

[20] D Pratas, M Hosseini, and A J Pinho, "GeCo2: An Optimized Tool for Lossless Compression and Analysis of DNA Sequences," in *International Conference on Practical Applications of Computational Biology & Bioinformatics*. Springer, 2019, pp. 137–145.

[21] M Silva, D Pratas, and A J Pinho, "Efficient DNA sequence compression with neural networks," *GigaScience*, vol. 9, no. 11, pp. giaa119, 2020.

[22] R Wang, Y Bai, Y Chu, Z Wang, Y Wang, and et al, "DeepDNA: a hybrid convolutional and recurrent neural network for compressing human mitochondrial genomes," in *2018 IEEE International Conference on Bioinformatics and Biomedicine (BIBM)*, 2018, pp. 270–274.

[23] D Pratas, M Hosseini, J M Silva, and A J Pinho, "A Reference-Free Lossless Compression Algorithm for DNA Sequences Using a Competitive Prediction of Two Classes of Weighted Models," *Entropy*, vol. 21, no. 11, pp. 1074, 2019.

[24] M H Mohammed, A Dutta, T Bose, S Chadaram, and S S Mande, "DELIMINATE—a fast and efficient method for loss-less compression of genomic sequences: sequence analysis," *Bioinformatics*, vol. 28, no. 19, pp. 2527–2529, 2012.

[25] A J Pinho and D Pratas, "MFCompress: a compression tool for FASTA and multi-FASTA data," *Bioinformatics*, vol. 30, no. 1, pp. 117–118, 2014.

[26] K Kryukov, M T Ueda, S Nakagawa, and T Imanishi, "Nucleotide Archival Format (NAF) enables efficient lossless reference-free compression of DNA sequences," *Bioinformatics*, vol. 35, no. 19, pp. 3826–3828, 2019.

[27] S Grabowski and T M Kowalski, "MBGC: Multiple Bacteria Genome Compressor," *GigaScience*, vol. 11, 2022.

[28] J M Silva, D Pratas, T Caetano, and S Matos, "The complexity landscape of viral genomes," *GigaScience*, vol. 11, 2022.

[29] M Hosseini, D Pratas, B Morgenstern, and A J Pinho, "Smash++: an alignment-free and memory-efficient tool to find genomic rearrangements," *GigaScience*, vol. 9, no. 5, pp. giaa048, 2020.

[30] D Pratas, M Hosseini, G Grilo, A J Pinho, R M Silva, et al., "Metagenomic composition analysis of an ancient sequenced polar bear jawbone from Svalbard," *Genes*, vol. 9, no. 9, pp. 445, 2018.

[31] D Pratas, M Toppinen, L Pyöriä, K Hedman, A Sajantila, and M F Perdomo, "A hybrid pipeline for reconstruction and analysis of viral genomes at multi-organ level," *GigaScience*, vol. 9, no. 8, pp. giaa086, 2020.

[32] A Moffat, R M Neal, and I H Witten, "Arithmetic coding revisited," *ACM Transactions on Information Systems (TOIS)*, vol. 16, no. 3, pp. 256–294, 1998.

[33] D Pratas, M Hosseini, and A J Pinho, "Substitutional tolerant Markov models for relative compression of DNA sequences," in *International Conference on Practical Applications of Computational Biology & Bioinformatics*. Springer, 2017, pp. 265–272.

[34] A J Pinho, D Pratas, and S P Garcia, "GReEn: a tool for efficient compression of genome resequencing data," *Nucleic Acids Research*, vol. 40, no. 4, pp. e27–e27, 2012.

[35] D Pratas, M Hosseini, J M Silva, and A J Pinho, "A reference-free lossless compression algorithm for dna sequences using a competitive prediction of two classes of weighted models," *Entropy*, vol. 21, no. 11, pp. 1074, 2019.

[36] M Hiransha, E A Gopalakrishnan, V K Menon, and K P Soman, "NSE stock market prediction using deep-learning models," *Procedia Computer Science*, vol. 132, pp. 1351–1362, 2018.

[37] M Silva, D Pratas, and A J Pinho, "AC2: An Efficient Protein Sequence Compression Tool Using Artificial Neural Networks and Cache-Hash Models," *Entropy*, vol. 23, no. 5, pp. 530, 2021.

[38] H Robbins and S Monro, "A stochastic approximation method," *The Annals of Mathematical Statistics*, pp. 400–407, 1951.

[39] S Nurk, S Koren, A Rhie, M Rautiainen, A V Bzikadze, et al., "The complete sequence of a human genome," *Science*, vol. 376, no. 6588, pp. 44–53, 2022.