DOI
Advance Access Publication Date: Day Month Year
Manuscript Category

Sequence analysis

Impact of lossy compression of nanopore raw signal data on basecall and consensus accuracy

Shubham Chandak *, Kedar Tatwawadi, Srivatsan Sridhar and Tsachy Weissman *

[1]Department of Electrical Engineering, Stanford University, Stanford, CA 94305, USA.

*To whom correspondence should be addressed.

Associate Editor: XXXXXXX

Received on XXXXX; revised on XXXXX; accepted on XXXXX

Abstract

Motivation: Nanopore sequencing provides a real-time and portable solution to genomic sequencing, with long reads enabling better assembly and structural variant discovery than second generation technologies. The nanopore sequencing process generates huge amounts of data in the form of raw current data, which must be compressed to enable efficient storage and transfer. Since the raw current data is inherently noisy, lossy compression has potential to significantly reduce space requirements without adversely impacting performance of downstream applications.

Results: We study the impact of two state-of-the-art lossy time-series compressors applied to nanopore raw current data, evaluating the tradeoff between compressed size and basecalling/consensus accuracy. We test several basecallers and consensus tools on a variety of datasets at varying depths of coverage, and conclude that lossy compression can provide as much as 40-50% reduction in compressed size of raw signal data with negligible impact on basecalling and consensus accuracy.

Availability: The code, data and supplementary results are available at `https://github.com/shubhamchandak94/lossy_compression_evaluation`.

Contact: schandak@stanford.edu

Keywords: nanopore raw signal, lossy compression, basecalling accuracy, consensus accuracy

1 Introduction

Nanopore sequencing technologies developed over the past decade provide a real-time and portable sequencing platform capable of producing long reads, with important applications in completing genome assemblies and discovering structural variants associated with several diseases [1]. Nanopore sequencing consists of a membrane with pores where DNA passes through the pore leading to variations in current passing through the pore. This electrical current signal is sampled to generate the raw signal data for the nanopore sequencer and is then basecalled to produce the read sequence. Due to the continuous nature of the raw signal and high sampling rate, the raw signal data requires large amount of space for storage, e.g., a typical human sequencing experiment can produce terabytes of raw signal data, which is an order of magnitude more than the space required for storing the basecalled reads [2].

Due to the ongoing research into improving basecalling technologies and the scope for further improvement in accuracy, the raw data needs to be retained to allow repeated analysis of the sequencing data. This makes compression of the raw signal data crucial for efficient storage and transport. There have been a couple of lossless compressors designed for nanopore raw signal data, namely, Picopore [3] and VBZ [4]. Picopore simply applies gzip compression to the raw data, while VBZ, which is the current state-of-the-art tool, uses variable byte integer encoding followed by zstd compression. Although VBZ reduces the size of the raw data by 60%, the compressed size is still quite significant and further reduction is desirable. However, obtaining further improvements in lossless compression is challenging due to the inherently noisy nature of the current measurements.

In this context, lossy compression is a natural candidate to provide a boost in compression at the cost of certain amount of distortion in the raw signal. There have been several works on lossy compression for time series data, including SZ [5, 6, 7] and LFZip [8] that provide a guarantee that the reconstruction lies within a certain user-defined interval of the original value for all time steps. However, in the case of nanopore raw current signal, the metric of interest is not the maximum deviation from the original value, but instead the impact on the performance of basecalling and other downstream analysis steps. In particular, two quantities of interest are the basecall accuracy and the consensus accuracy. The basecall accuracy measures the similarity of the basecalled read sequence to the known true

*Shubham Chandak, Kedar Tatwawadi and Srivatsan Sridhar are PhD students in Electrical Engineering at Stanford University. Tsachy Weissman is a Professor of Electrical Engineering at Stanford University.

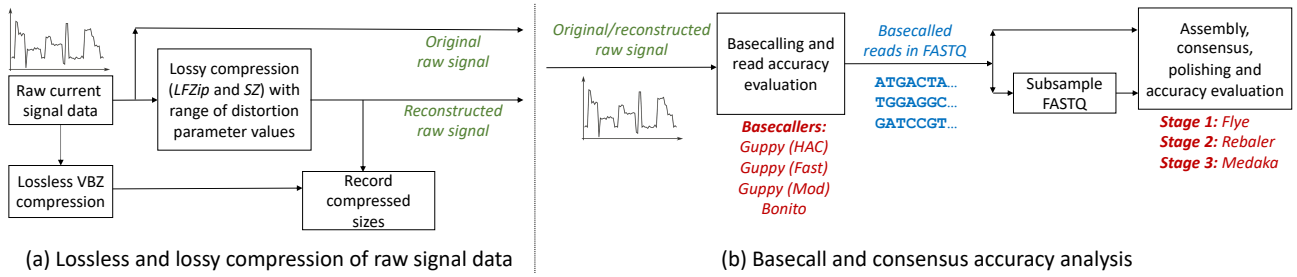

Fig. 1. Flowchart showing the experimental procedure. (a) The raw data was compressed with both lossless and lossy compression tools, (b) the original and lossily compressed data was then basecalled with four basecalling tools. Finally, the basecalled data and its subsampled versions were assembled and the assembly (consensus) was polished using a three-step pipeline (1. Flye, 2. Rebaler, 3. Medaka). The tradeoff between compressed size and basecalling/consensus accuracy was studied. Parts of the pipeline are based on previous work in [9] and [10].

genome sequence, while the consensus accuracy measures the similarity of the consensus sequence obtained from multiple overlapping reads to the known true genome sequence. As discussed in [9], these two measures are generally correlated but can follow different trends in the presence of systematic errors. In general, consensus accuracy can be thought of as the higher-level metric which is usually of interest in most applications, while basecall accuracy is a lower-level metric in the sequencing analysis pipeline.

In this work, we study the impact of lossy compression of nanopore raw signal data on basecall and consensus accuracy. We evaluate the results for several basecallers and at multiple stages of the consensus pipeline to ensure the results are generalizable to future iterations of these tools. We find that lossy compression using general-purpose tools can provide significant reduction in file sizes with negligible impact on accuracy. We believe our results provide motivation for research into specialized lossy compressors for nanopore raw signal data and suggest the possibility of reducing the resolution of the raw signal generated on the nanopore device itself while preserving the accuracy. The source code for our analysis is publicly available and can be useful as a benchmarking pipeline for further research into lossy compression for nanopore data.

2 Background

2.1 Nanopore sequencing and basecalling

Nanopore sequencing, specifically the MinION sequencer developed by Oxford Nanopore Technologies (ONT) [1], involves a strand of DNA passing through a pore in a membrane with a potential applied across it. Depending on the sequence of bases present in the pore (roughly 6 bases), the ionic current passing through the pore varies with time and is measured at a sampling frequency of 4 kHz. The sequencing produces 5-15 current samples per base, which are quantized to a 16-bit integer and stored as an array in a version of the HDF5 format called fast5. The current signal is then processed by the basecaller to generate the basecalled read sequence and the corresponding quality value information. In the uncompressed format, the raw current signal requires close to 18 bytes per sequenced base which is significantly more than the amount needed for storing the sequenced base and the associated quality value.

Over the past years, there has been a shift in the basecalling strategy from a physical model-based approach to a machine learning-based approach leading to significant improvement in basecalling accuracy (see [11] and [9] for a detailed review). In particular, the current default basecaller Guppy by ONT (based on open source tool Flappie [12]) uses a recurrent neural network that generates transition probabilities for the bases which are then converted to the most probable sequence of bases using Viterbi algorithm. Another recent basecaller by ONT is bonito [13] (currently experimental), which is based on a QuartzNet convolutional neural network [14] and CTC decoding [15], achieving close to 92-95% basecalling accuracy in terms of edit distance (see Figure 3). Despite the progress in basecalling, the current error rates are still relatively high with considerable fraction of insertion and deletion errors, which necessitates the storage of the raw current data for utilizing improvements in the basecalling technologies for future (re)analysis.

2.2 Assembly, consensus and polishing

Long nanopore reads allow much better repeat resolution and are able to capture long-range information about the genome leading to significant improvements in de novo genome assembly [2]. However genome assembly with nanopore data needs to handle the much higher error rates as compared to second generation technologies such as Illumina, and there have been several specialized assemblers for this purpose, including Flye [16, 17] and Canu [18], some of which allow hybrid assembly with a combination of short and long read data.

Nanopore de novo assembly is usually followed by a consensus step that improves the assembly quality by aligning the reads to a draft assembly and then performing consensus from overlapping reads (e.g., Racon [19]). Note that the consensus step can be performed even without de novo assembly if a reference sequence for the species is already available, in which case the alignment to the reference is used to determine the overlap between reads. Further polishing of the consensus sequence can be performed with tools specialized for nanopore sequencing that use the noise characteristics of the sequencing and/or basecalling to find the most probable consensus sequence. For example, Nanopolish [20] directly uses the raw signal data for polishing the consensus using a probabilistic model for the generation of the raw signal from the genomic sequence. Medaka [21] by ONT is the current state-of-the-art consensus polishing tool both in terms of runtime and accuracy [10]. Medaka uses a neural network to perform the consensus from the pileup of the basecalled reads at each position of the genome.

2.3 Lossy compression

Lossy compression [22] refers to compression of the data into a compressed bitstream where the decompressed (reconstructed) data need not be exactly but only approximately similar to the original data. Lossy compression is usually studied in the context of a distortion metric that specifies how the distortion or error between the original and the reconstruction is measured. This gives rise to a tradeoff between the compressed size and the distortion, referred to as the rate-distortion curve. Here, we work with two state-of-the-art lossy compressors for time-series data, LFZip [8] and SZ [5, 6, 7]. Both these compressors work with a maxerror parameter that specifies the maximum absolute deviation between the original and the reconstructed data. If $x_1, \ldots, x_T$ denotes the original time-series, $\hat{x}_1, \ldots, \hat{x}_T$ denotes the reconstructed time-series, and ϵ is the maxerror parameter, then these

Species	Sample	Reference chrom. size (bp)	GC-content	Flowcell type	Read count	Read length N50 (bp)	Approx. depth/coverage	Raw signal size (GB)	
								Uncompressed	VBZ (lossless)
Staphylococcus aureus	CAS38_02	2,902,076	32.8%	R9.4.1	11,047	24,666	83x	4.86	2.02
Klebsiella pneumoniae	INF032	5,111,537	57.6%	R9.4	15,154	37,181	108x	10.14	4.32

Table 1. Datasets used for analysis, obtained from [9]. N50 is a statistical measure of average length of the reads (see [9] for a precise definition). The uncompressed size column refers to storing the raw signal in the default representation using 16 bits/signal value.

compressors guarantee that $\max_{t=1,\ldots,T} |x_t - \hat{x}_t| < \epsilon$. LFZip and SZ use slightly different approaches towards lossy compression, LFZip uses a prediction-quantization-entropy coding approach, while SZ uses a curve fitting step followed by entropy coding.

There are a couple of reasons for focusing on lossy compressors with maximum absolute deviation as the distortion metric instead of mean square error or mean absolute error in this work. The first reason is that the guaranteed maximum error implies that the reconstructed raw signal is close to the original value at each and every timestep and not only in the average sense. Hence, the maximum error distortion metric is preferable for general applications where the true distortion metric is not well understood. The second reason is the availability of efficient implementations which is crucial for compressing the large genomic datasets. However, we believe that there is significant scope for using mean square error and other metrics for developing specialized lossy compressors for nanopore data given the better theoretical understanding of those metrics.

3 Experiments

We next describe the experimental setup in detail (see Figure 1 for a flowchart representation). All the scripts are available on the GitHub repository. The experiments were run on an Ubuntu 18.04.4 server with 40 Intel Xeon processors (2.2 GHz), 260 GB RAM and 8 Nvidia TITAN X (Pascal) GPUs.

3.1 Datasets

Table 1 shows the bacterial datasets used for analysis in this work. These high quality datasets were obtained from [9] and were chosen to be representative datasets with different GC-content and flowcell type. For both datasets the ground-truth genomic sequence is known through hybrid assembly with long and short read technologies. The table also shows the uncompressed and VBZ compressed sizes for the two datasets, we observe that lossless compression can provide size reduction of close to 60%. We should note that while we work with relatively small bacterial genomes for ease of experimentation, our analysis generalizes to larger genomes due to the local nature of the lossy compression as well as the basecalling and consensus pipeline.

For each dataset, we run our analysis on both the original read depth (83x and 108x) as well as subsampled versions of the datasets (2x, 4x and 8x subsampling of Fastq files performed using Seqtk [23]). This helps us understand how the impact of lossy compression on consensus accuracy depends on read depth.

3.2 Lossy compression

To study the impact of lossy compression, we generate new fast5 datasets by replacing the raw signal in the original fast5 files with the reconstruction produced by lossy compression. We use open source general-purpose time-series compressors LFZip [8] (version 1.1) and SZ [5, 6, 7] (version 2.1.8). Both the tools require a parameter representing the maximum absolute deviation (maxerror) of the reconstruction from the original. We conducted ten experiments for each tool by setting the maxerror parameter to $1, 2, \ldots, 10$. To put this in context, note that for a typical current range of 60 pA and the typical noise value of 1-2 pA, the maxerror settings of 1 and 10 correspond to current values of 0.17 pA and 1.7 pA, respectively [24]. Finally, we note that both LFZip and SZ can compress millions of timesteps per second and hence can be used to compress the nanopore raw signal data in real time as it is produced by the sequencer. Note that while LFZip has two prediction modes, we use the default NLMS (linear) predictor mode since the neural network predictor is quite slow and currently not suitable for large genomic datasets.

3.3 Basecalling and consensus

We perform basecalling on the raw signal data (both original and lossily compressed) using three modes of Guppy (version 3.4.5) as well as with bonito (version 0.0.5, note that bonito is currently an experimental release). For Guppy, we use the default high accuracy mode (guppy_hac), the fast mode (guppy_fast) and the mode trained on native DNA with 5mC and 6mA modified bases (guppy_mod). All the modes use the same general framework but differ in terms of the neural network architecture size and weights. We use these four basecaller settings to study whether lossy compression leads to loss of useful information that can be potentially exploited by future basecallers.

We use a three-step assembly, consensus and polishing pipeline based on the analysis and recommendations in [10]. The first step is de novo assembly using Flye (version 2.7-b1585) [16, 17] which produces a basic draft assembly. The second step is consensus polishing of the Flye assembly using Rebaler (version v0.2.0) which runs multiple rounds of Racon (version 1.4.3) to produce a high quality consensus of the reads. Finally, the third step uses Medaka (version 0.11.5) by ONT that performs further polishing of the Rebaler consensus using a neural network based approach. Note that the neural network model for Medaka needs to be chosen corresponding to the basecaller.

3.4 Evaluation metrics

For evaluating the basecalling and consensus accuracy, we use the pipeline presented in [9] and [10]. The basecalled reads were aligned to the true genome sequence using minimap2 [25] and the read's basecalled identity was defined as the number of matching bases in the alignment divided by the total alignment length. We report only the median identity across reads in the results section, while the complete results are available on GitHub. The consensus accuracy after each stage is computed in a similar manner, where instead of aligning the reads, we split the assembly into 10 kbp pieces and then find median identity across these pieces. Finally, we compute the basecall and consensus Qscore using the Phred scale as $\text{Qscore} = -10\log_{10}(1 - \text{identity})$ where the identity is represented as a fraction. We refer the reader to [9] for further discussion on these metrics.

4 Results and discussion

We now discuss the main results obtained from the experiments described above. Throughout the results and discussion, the compressed sizes for lossy compression are shown relative to the compressed size for VBZ

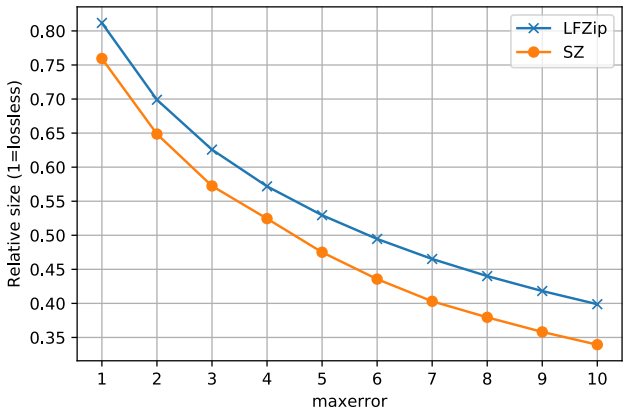

Fig. 2. Compressed size for lossy compression with LFZip and SZ for the S. aureus dataset as a function of the maxerror parameter. The compressed sizes are shown relative to the VBZ lossless compression size.

lossless compression. Detailed supplementary results and additional plots are available on the GitHub repository.

Figure 2 shows the variation of the size of the lossy compressed dataset with the maxerror parameter. We see that SZ generally provides better compression than LFZip at the same maxerror value. We also see that lossy compression can provide significant size reduction over lossless compression even with relatively small maxerror (recall that maxerror of 1 in the 16-bit representation of the raw signal corresponds to 0.17 pA error in the current value). For example, at maxerror of 5, lossy compression can provide size reduction of around 50% over lossless compression and size reduction of around 70% over the uncompressed 16-bit representation. Both SZ and LFZip can compress millions of samples per second, with SZ being about an order of magnitude faster than LFZip [8].

Figure 3 shows the tradeoff achieved between basecalling accuracy and compressed size for the two datasets for all four basecallers. First, we observe that bonito performs the best, followed by guppy_hac, guppy_mod and guppy_fast. As the maxerror parameter is increased, the basecalling accuracy stays stable for all the basecallers till the compressed size reaches 65% of the losslessly compressed size. After this the basecalling accuracy drops more sharply, becoming 2% lower than the original lossless level when the maxerror parameter is 10. The drop seems to follow a similar trend for all the basecallers and compressors, suggesting that at least 35% reduction in size over lossless compression can be obtained without sacrificing basecalling accuracy. Note that for maxerror parameter equal to 10, the allowed deviation of the reconstructed raw signal is larger than the typical noise levels in sequencing, and hence lossy compression probably leads to perceptible loss in the useful information contained in the raw signal.

We note that the impact of lossy compression on read lengths and the number of aligned reads is negligible (supplementary results). This suggests that lossy compression generally leads to local and small perturbations in the basecalled read and does not lead to major structural changes in the read such as loss of information due to trimmed/shortened reads. This is expected given that the lossy compressors used here guarantee that the reconstructed signal is within a certain deviation from the original signal at each time step.

Figures 4, 5 and 6 study the tradeoff between consensus accuracy and compressed size (i) across basecallers for the final Medaka polished assembly, (ii) across the assembly stages and (iii) across read depths in subsampled datasets, respectively. As expected, we observe that the consensus accuracy is significantly higher than basecall accuracy across these experiments. We also observe that the consensus accuracy stays at the original lossless level till the compressed size reaches around 40-50% of the losslessly compressed size (50-60% reduction), and drop in accuracy beyond this is relatively small. Thus, the impact of lossy compression on consensus accuracy is less severe than that on basecall accuracy. This suggests that the errors introduced by lossy compression are generally random in nature and are mostly corrected by the consensus process. We note that lossy compression does not affect the number of assembled contigs (always 1) and the contig length (supplementary results), which again suggests that the impact of lossy compression is localized and there is no large-scale disruption in the assembly/consensus process.

Figure 5 considers the consensus accuracy after each stage of the assembly/consensus process (Flye, Rebaler, Medaka) for the *S. aureus* dataset basecalled with bonito. We see that each stage leads to further improvement in the consensus accuracy. We also observe that the earlier stages of the pipeline are impacted more heavily by lossy compression (in terms of percentage reduction in accuracy) than the final Medaka stage. This is expected since each successive stage of the assembly/consensus pipeline provides further correction of the basecalling errors caused due to lossy compression. This effect is similar to the equalizing effect of polishing applied to different basecallers observed in [9]. We see a similar trend for the other dataset and analysis tools (see supplementary results).

Finally, Figure 6 studies the impact of subsampling to lower read depths on the consensus accuracy (after Medaka polishing) for the *S. aureus* dataset basecalled with bonito. Note that the original dataset has around 80x depth of coverage, so the 8x subsampling produces a 10x depth which is generallly considered quite low. We observe that lossy compression has more severe impact on consensus accuracy for lower depths, but 40-50% of size reduction can still be achieved without sacrificing the accuracy. This is again expected because consensus works better with higher depth datasets and is able to correct a greater fraction of the basecalling errors. We see a similar trend for the other dataset and analysis tools (see supplementary results).

Overall, we observe that both LFZip and SZ can be used as tools to significantly save on space without sacrificing basecalling and consensus accuracy. The savings in space are close to 50% over lossless compression and 70% over the uncompressed representation. While it is not possible to say with certainty that we don't lose any information in the raw signal that might be utilized by future basecallers, the results for the different basecallers and consensus stages suggest that applying lossy compression (for a certain range of parameters) only affects the noise in the raw signal without affecting the useful components. Finally, the decision to apply lossy compression and the extent of lossy compression should be based on the read depth (coverage), with more savings possible at higher depths where consensus accuracy is the metric of interest.

5 Conclusions and future work

In this work, we explore the impact of lossy compression of nanopore raw current data on the basecall and consensus accuracy and find that lossy compression with existing tools can reduce the compressed size by 40-50% over lossless compression with negligible impact on accuracy. This conclusion holds across datasets at different depths of coverage as well as several basecalling and assembly tools, suggesting that lossy compression with appropriate parameters does not lead to loss of useful information in the raw signal. For datasets with high depth of coverage, even further reduction is possible without sacrificing consensus accuracy. The analysis pipeline and data, mostly based on [9] and [10], are available online on GitHub and can be useful for further experimentation and development of specialized lossy compressors for nanopore raw signal data, which is part of future work.

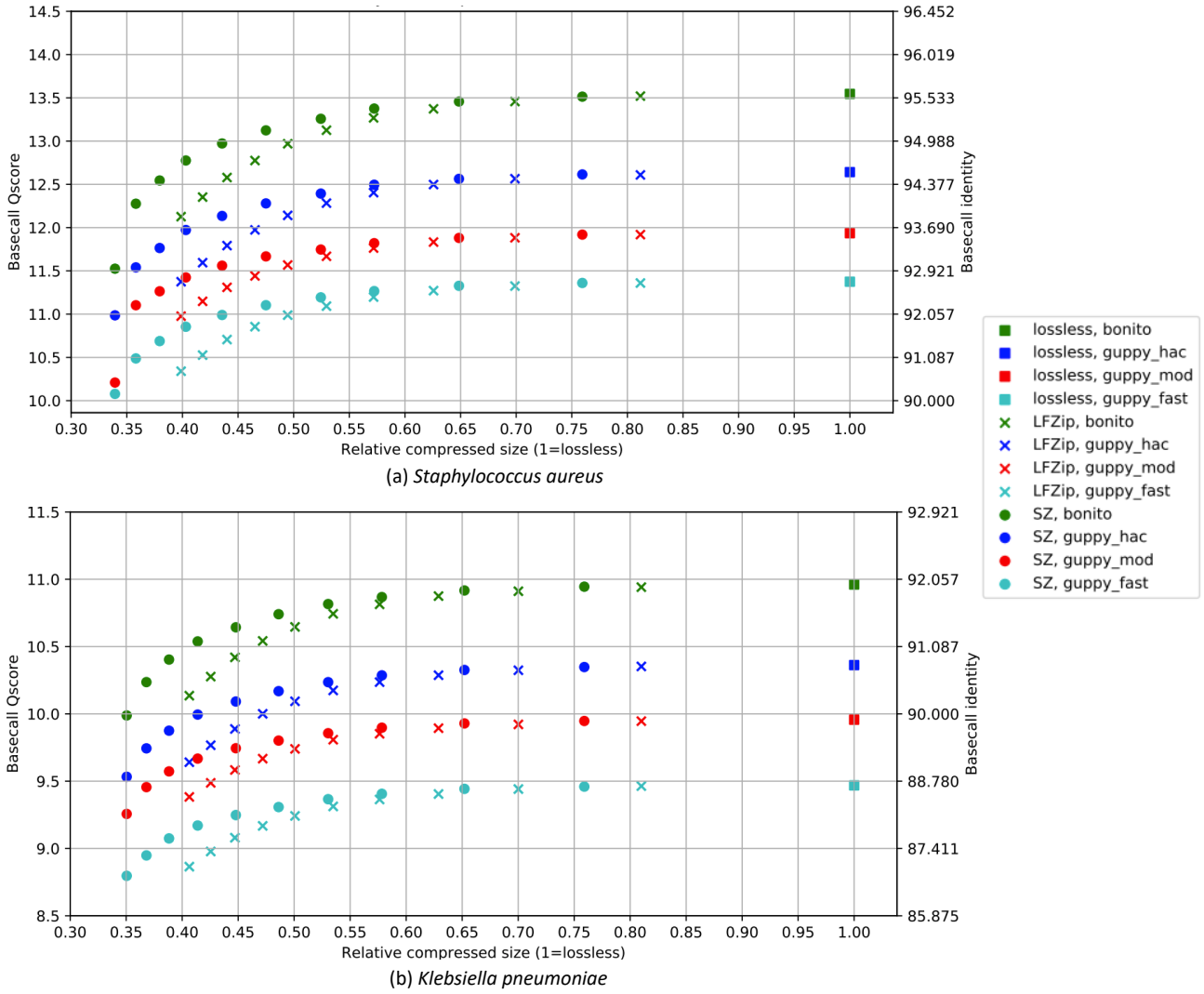

Fig. 3. Basecalling accuracy vs. compressed size for (a) S. aureus and (b) K. pneumoniae datasets. The results are displayed for the losslessly compressed data and the lossy compressed versions with LFZip and SZ (with maxerror 1 to 10) for the four basecallers. The compressed sizes are shown relative to the VBZ lossless compression size.

We believe that further research in this direction can lead to lossy compression algorithms tuned to the specific structure of the nanopore data and the evaluation metrics of interest, leading to further reduction in the compressed size. Further research into modeling the raw signal and the noise characteristics can help in this front. Another interesting direction could be the possibility of jointly designing the lossy compression with the modification of the algorithms in the downstream applications to match this compression. Finally, just as research on lossy compression for Illumina quality scores led to Illumina reducing the default resolution of quality scores [26], it might be interesting to explore a similar possibility for nanopore data by performing the quantization/lossy compression on the nanopore sequencing device itself.

Key points:

- Raw current signal data for nanopore sequencing requires large amounts of storage space and is difficult to losslessly compress due to noise.
- We evaluate the impact of lossy compression of nanopore raw signal data on basecall and consensus accuracy.
- We find that lossy compression with general-purpose tools can provide more than 40% reduction in space over lossless compression with negligible impact on accuracy.
- The extent of lossy compression should be determined based on the depth of coverage, with greater savings possible at higher depths.
- Further research on specialized lossy compression for nanopore raw signal data can provide a better tradeoff between compressed size and accuracy.

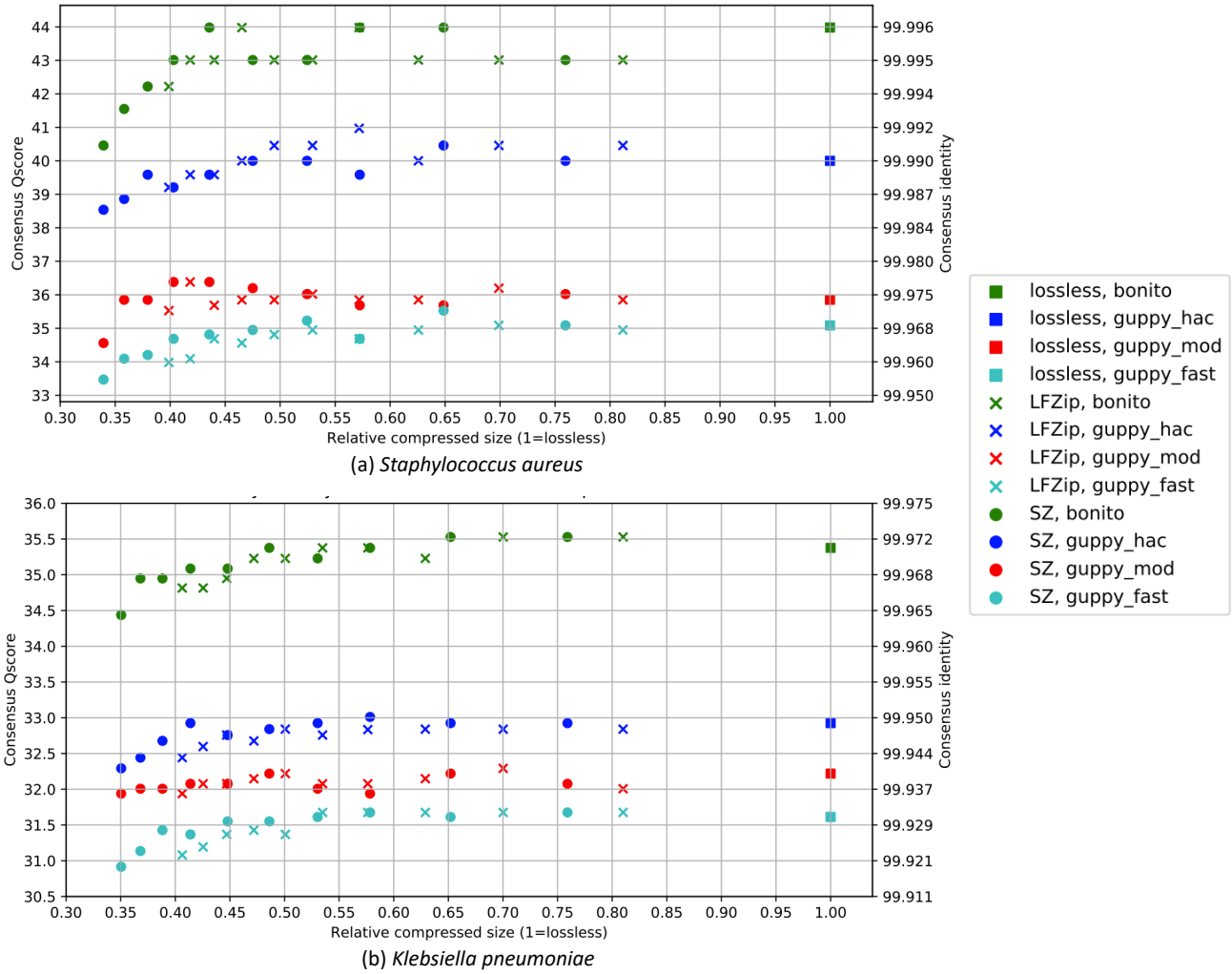

Fig. 4. Consensus accuracy vs. compressed size for (a) S. aureus and (b) K. pneumoniae datasets. The results are displayed for the polished Medaka assembly for the losslessly compressed data and the lossy compressed versions with LFZip and SZ (with maxerror 1 to 10) for the four basecallers. The compressed sizes are shown relative to the VBZ lossless compression size.

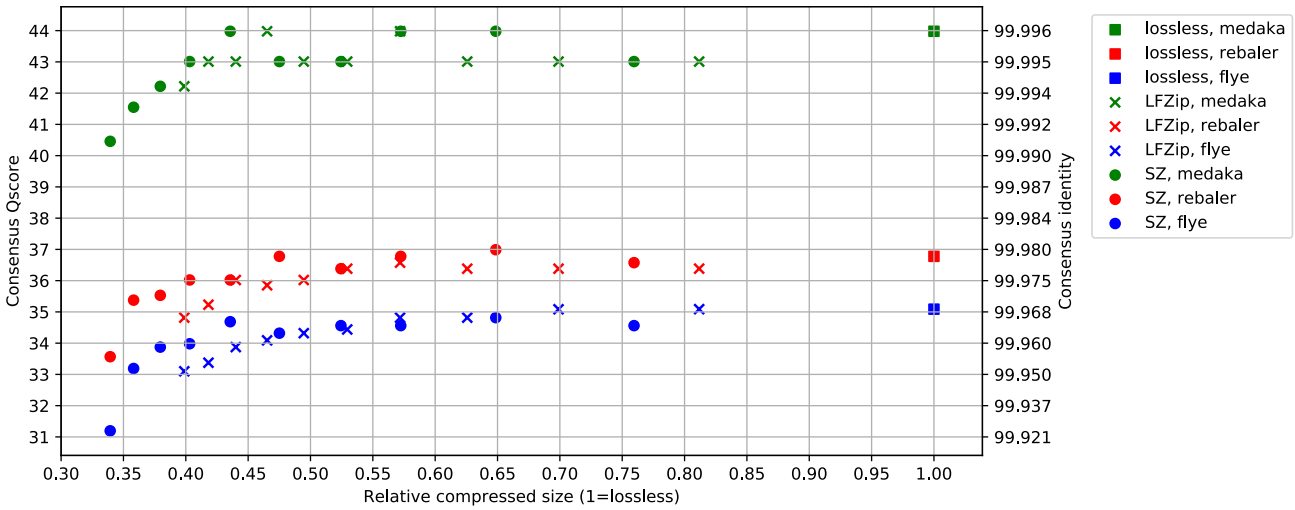

Fig. 5. Consensus accuracy vs. compressed size after each assembly step (Flye, Rebaler, Medaka) for the S. aureus dataset and bonito basecaller. The results are displayed for the losslessly compressed data and the lossy compressed versions with LFZip and SZ (with maxerror 1 to 10). The compressed sizes are shown relative to the VBZ lossless compression size.

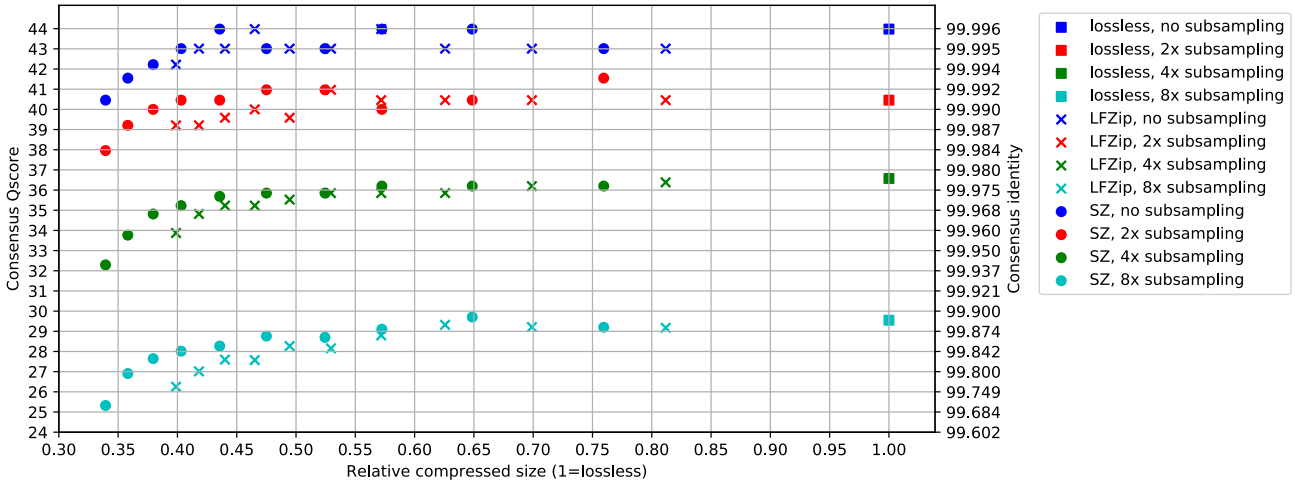

Fig. 6. Consensus accuracy vs. compressed size for subsampled versions (original, 2X subsampled, 4X subsampled, 8X subsampled) of the S. aureus dataset basecalled with bonito. The results are displayed for the polished Medaka assembly for the losslessly compressed data and the lossy compressed versions with LFZip and SZ (with maxerror 1 to 10). The compressed sizes are shown relative to the VBZ lossless compression size.

Funding

We acknowledge support from NSF Center for Science of Information, Siemens, Philips, and NIH.

References

[1] Miten Jain, Hugh E Olsen, Benedict Paten, and Mark Akeson. The Oxford Nanopore MinION: delivery of nanopore sequencing to the genomics community. *Genome biology*, 17(1):239, 2016.

[2] Miten Jain, Sergey Koren, Karen H Miga, Josh Quick, Arthur C Rand, Thomas A Sasani, John R Tyson, Andrew D Beggs, Alexander T Dilthey, Ian T Fiddes, et al. Nanopore sequencing and assembly of a human genome with ultra-long reads. *Nature biotechnology*, 36(4):338, 2018.

[3] Scott Gigante. Picopore: a tool for reducing the storage size of oxford nanopore technologies datasets without loss of functionality. *F1000Research*, 6, 2017.

[4] VBZ Compression. https://github.com/nanoporetech/vbz_compression/. Last accessed: March 27, 2020.

[5] Sheng Di and Franck Cappello. Fast error-bounded lossy HPC data compression with SZ. In *2016 ieee international parallel and distributed processing symposium (ipdps)*, pages 730–739. IEEE, 2016.

[6] Dingwen Tao, Sheng Di, Zizhong Chen, and Franck Cappello. Significantly improving lossy compression for scientific data sets based on multidimensional prediction and error-controlled quantization. In *2017 IEEE International Parallel and Distributed Processing Symposium (IPDPS)*, pages 1129–1139. IEEE, 2017.

[7] Xin Liang, Sheng Di, Dingwen Tao, Sihuan Li, Shaomeng Li, Hanqi Guo, Zizhong Chen, and Franck Cappello. Error-controlled lossy compression optimized for high compression ratios of scientific datasets. In *2018 IEEE International Conference on Big Data (Big Data)*, pages 438–447. IEEE, 2018.

[8] Shubham Chandak, Kedar Tatwawadi, Chengtao Wen, Lingyun Wang, Juan Aparicio, and Tsachy Weissman. LFZip: Lossy compression of multivariate floating-point time series data via improved prediction. *arXiv preprint arXiv:1911.00208*, 2019.

[9] Ryan R Wick, Louise M Judd, and Kathryn E Holt. Performance of neural network basecalling tools for Oxford Nanopore sequencing. *Genome biology*, 20(1):129, 2019.

[10] Ryan R Wick, Louise M Judd, and Kathryn E Holt. August 2019 consensus accuracy update. https://github.com/rrwick/August-2019-consensus-accuracy-update/. 2019. Last accessed: March 27, 2020.

[11] Franka J Rang, Wigard P Kloosterman, and Jeroen de Ridder. From squiggle to basepair: computational approaches for improving nanopore sequencing read accuracy. *Genome biology*, 19(1):90, 2018.

[12] Flappie: Flip-flop basecaller for Oxford Nanopore reads. https://github.com/nanoporetech/flappie. Last accessed: March 27, 2020.

[13] bonito. https://github.com/nanoporetech/bonito/. Last accessed: March 27, 2020.

[14] Samuel Kriman, Stanislav Beliaev, Boris Ginsburg, Jocelyn Huang, Oleksii Kuchaiev, Vitaly Lavrukhin, Ryan Leary, Jason Li, and Yang Zhang. Quartznet: Deep automatic speech recognition with 1d time-channel separable convolutions. *arXiv preprint arXiv:1910.10261*, 2019.

[15] Alex Graves, Santiago Fernández, Faustino Gomez, and Jürgen Schmidhuber. Connectionist temporal classification: labelling unsegmented sequence data with recurrent neural networks. In *Proceedings of the 23rd international conference on Machine learning*, pages 369–376, 2006.

[16] Mikhail Kolmogorov, Jeffrey Yuan, Yu Lin, and Pavel A Pevzner. Assembly of long, error-prone reads using repeat graphs. *Nature biotechnology*, 37(5):540–546, 2019.

[17] Yu Lin, Jeffrey Yuan, Mikhail Kolmogorov, Max W Shen, Mark Chaisson, and Pavel A Pevzner. Assembly of long error-prone reads using de Bruijn graphs. *Proceedings of the National Academy of Sciences*, 113(52):E8396–E8405, 2016.

[18] Sergey Koren, Brian P Walenz, Konstantin Berlin, Jason R Miller, Nicholas H Bergman, and Adam M Phillippy. Canu: scalable and accurate long-read assembly via adaptive k-mer weighting and repeat separation. *Genome research*, 27(5):722–736, 2017.

[19] Robert Vaser, Ivan Sović, Niranjan Nagarajan, and Mile Šikić. Fast and accurate de novo genome assembly from long uncorrected reads. *Genome research*, 27(5):737–746, 2017.

[20] Nicholas J Loman, Joshua Quick, and Jared T Simpson. A complete bacterial genome assembled de novo using only nanopore sequencing data. *Nature methods*, 12(8):733–735, 2015.

[21] Medaka. https://nanoporetech.github.io/medaka/. Last accessed: March 27, 2020.

[22] Allen Gersho and Robert M Gray. *Vector quantization and signal compression*, volume 159. Springer Science & Business Media, 2012.

[23] Heng Li. Seqtk. https://github.com/lh3/seqtk/. Last accessed: March 27, 2020.

[24] Nanopore k-mer model. https://github.com/nanoporetech/kmer_models/. Last accessed: March 27, 2020.

[25] Heng Li. Minimap2: pairwise alignment for nucleotide sequences. *Bioinformatics*, 34(18):3094–3100, 2018.

[26] Illumina. NovaSeq 6000 System Quality Scores and RTA3 Software. https://www.illumina.com/content/dam/illumina-marketing/documents/products/appnotes/novaseq-hiseq-q30-app-note-770-2017-010.pdf. 2017.