

Restoring flowcell type and basecaller configuration from FASTQ files of nanopore sequencing data

Received: 1 December 2024

Accepted: 22 April 2025

Published online: 02 May 2025

Jun Mencius^{1,4}, Wenjun Chen^{2,4}, Youqi Zheng¹, Tingyi An¹, Yongguo Yu^{2,5}✉, Kun Sun^{2,3,5}✉, Huijuan Feng^{1,5}✉ & Zhixing Feng^{2,5}✉

As nanopore sequencing has been widely adopted, data accumulation has surged, resulting in over 700,000 public datasets. While these data hold immense potential for advancing genomic research, their utility is compromised by the absence of flowcell type and basecaller configuration in about 85% of the data and associated publications. These parameters are essential for many analysis algorithms, and their misapplication can lead to significant drops in performance. To address this issue, we present LongBow, designed to infer flowcell type and basecaller configuration directly from the base quality value patterns of FASTQ files. LongBow has been tested on 66 in-house basecalled FAST5/POD5 datasets and 1989 public FASTQ datasets, achieving accuracies of 95.33% and 91.45%, respectively. We demonstrate its utility by reanalyzing nanopore sequencing data from the COVID-19 Genomics UK (COG-UK) project. The results show that LongBow is essential for reproducing reported genomic variants and, through a LongBow-based analysis pipeline, we discovered substantially more functionally important variants while improving accuracy in lineage assignment. Overall, LongBow is poised to play a critical role in maximizing the utility of public nanopore sequencing data, while significantly enhancing the reproducibility of related research.

Oxford Nanopore Technologies (ONT) sequencing offers numerous advantages, including long read length, unbiased sequencing coverage, and direct base modification detection. These features significantly enhance variant calling, haplotype phasing, genome assembly, and epigenetic studies¹. As nanopore sequencing technology becomes more accurate and cost-effective², its adoption has surged in recent years, leading to the accumulation of over 700,000 datasets in the Sequence Read Archive (SRA) database (<https://www.ncbi.nlm.nih.gov/sra>). This wealth of data holds immense potential for advancing comparative genomics, population genetics, and the identification of pathogenic variants^{3–6}. However, reusing and integrating

ONT data for robust analyses, or even reproducing results from published studies based on ONT data, remains challenging. A primary issue is that many state-of-the-art computational tools for ONT data analysis require specific information about flowcell types and basecaller configurations (software, version, and mode) (Table 1 and Supplementary Data 1), which are missing in most public data. The gold standard for obtaining flowcell type is to extract such information from the raw FAST5/POD5 files of ONT data⁷. However, the vast majority of public ONT data do not include raw data files (Fig. 1a). The most convenient way to obtain flowcell type and basecaller configurations is to search the metadata table in the database, but this information is missing in

¹Department of Computational Biology, School of Life Sciences, Fudan University, Shanghai, China. ²Department of Clinical Genetics, Xinhua Hospital affiliated to Shanghai Jiao Tong University School of Medicine, Shanghai, China. ³Department of Pediatric Cardiology, Xinhua Hospital affiliated to Shanghai Jiao Tong University School of Medicine, Shanghai, China. ⁴These authors contributed equally: Jun Mencius, Wenjun Chen. ⁵These authors jointly supervised this work: Yongguo Yu, Kun Sun, Huijuan Feng, Zhixing Feng. ✉e-mail: yuyongguo@xinhuaamed.com.cn; sunkun@xinhuaamed.com.cn; huijuanfeng@fudan.edu.cn; fengzhixing@shsmu.edu.cn

Table 1 | List of ONT data analysis software with reliance on flowcell types or basecaller configurations

Application	Software	Flowcell type	Basecaller type	Basecaller version	Basecalling mode	Indirect
Sequence alignment	minimap2 ³⁹	●				
Error correction	HERRO ³⁹	●				
Variants calling	Clair3 ¹⁰	●	●	●	HAC/SUP ^[a]	
	DeepVariant ⁴⁰	●				
	PEPPER ⁴¹	●	●	●	SUP	
	Dysgu ⁴²	●				
	nanomonsv ⁴³	●	●	●	unspecified	
	Medaka ^[b]	●	●	●	FAST, HAC, SUP	
Haplotype phasing	WhatsHap ⁴⁴					● ^[c]
	Margin ⁴¹	●	●	●	unspecified	
	HapCut2 ⁴⁵					● ^[c]
	LongPhase ⁴⁶					● ^[c]
Genome assembly	Flye/MetaFlye ^{17,47}	●	●	●	HAC/SUP	
	Shasta ¹¹	●	●	●	HAC for Guppy 4, SUP for Guppy 6 or higher	
	Canu ⁴⁸	●				
	wtdbg2 ⁴⁹					● ^[d]
Genome polishing	MarginPolish ¹¹	●				
	HomoPolish ⁵⁰	●	●	●	FAST, HAC, SUP	

^[a] Clair3 uses the same model for the HAC and SUP modes of R9 Guppy6 data, but it uses different models for the HAC and SUP modes of R10 data.

^[b] <https://github.com/nanoporetech/medaka>.

^[c] These algorithms depend on the outputs of variant callers, which may be influenced by flowcell type or basecaller configuration.

^[d] Wtdbg2 depends on the sequencing error rate, which is influenced by flowcell type and basecaller configuration.

A circle mark in “Flowcell type”, “Basecaller type”, “Basecaller version”, or “Basecalling mode” indicates that the software explicitly requires this information as a mandatory input parameter. A circle mark in “Indirect” indicates that the software requires parameters derived from flowcell type or basecaller configuration, such as sequencing error rate or outputs from other software relying on these parameters. In “Basecalling mode”, “/” means the models might be used interchangeably, while “,” means each mode has a distinct associated model. If a mode is not listed, it means the current version of the software does not explicitly support it. “Unspecified” indicates a single model is provided for all modes, but it is not documented whether they can be used interchangeably. The references for the software are listed in the section References and cited in the “Software” field of the table.

96.94% of SRA records (Fig. 1b, details are in Methods). BAM/CRAM files might contain flowcell types and basecaller configurations in their headers, but these files are absent in most public ONT data (Supplementary Fig. 1). Additionally, a thorough manual review of 100 randomly selected ONT datasets from SRA reveals that only 15% (95% confidence interval is [9.31%, 23.28%]) of these reports include complete information on flowcell types and basecaller configurations in either the associated publications, SRA metadata, FAST5/POD5 files, BAM/CRAM files or other data sources. (Fig. 1c and Supplementary Data 2).

Flowcell type and basecaller configurations have a substantial impact on the sequencing accuracy and error patterns⁸. Employing parameters or pretrained models that do not match the flowcell types or basecaller configurations can drastically reduce algorithm performance (Fig. 1d–g). This discrepancy arises from the algorithms’ reliance on specific assumptions or pretrained models tailored to distinct flowcell types and basecaller configurations. Moreover, since these algorithms perform basic tasks in data analysis, such as sequence alignment, variant calling, and genome assembly, the downstream high-level analysis like molecular diagnosis based on detected pathogenic variants and population genomics studies based on assemblies are also affected by flowcell types or basecaller configurations indirectly. Therefore, the absence of this critical metadata poses a significant barrier to the effective integration and analysis of public ONT datasets and presents substantial challenges to the reproducibility of ONT-related research.

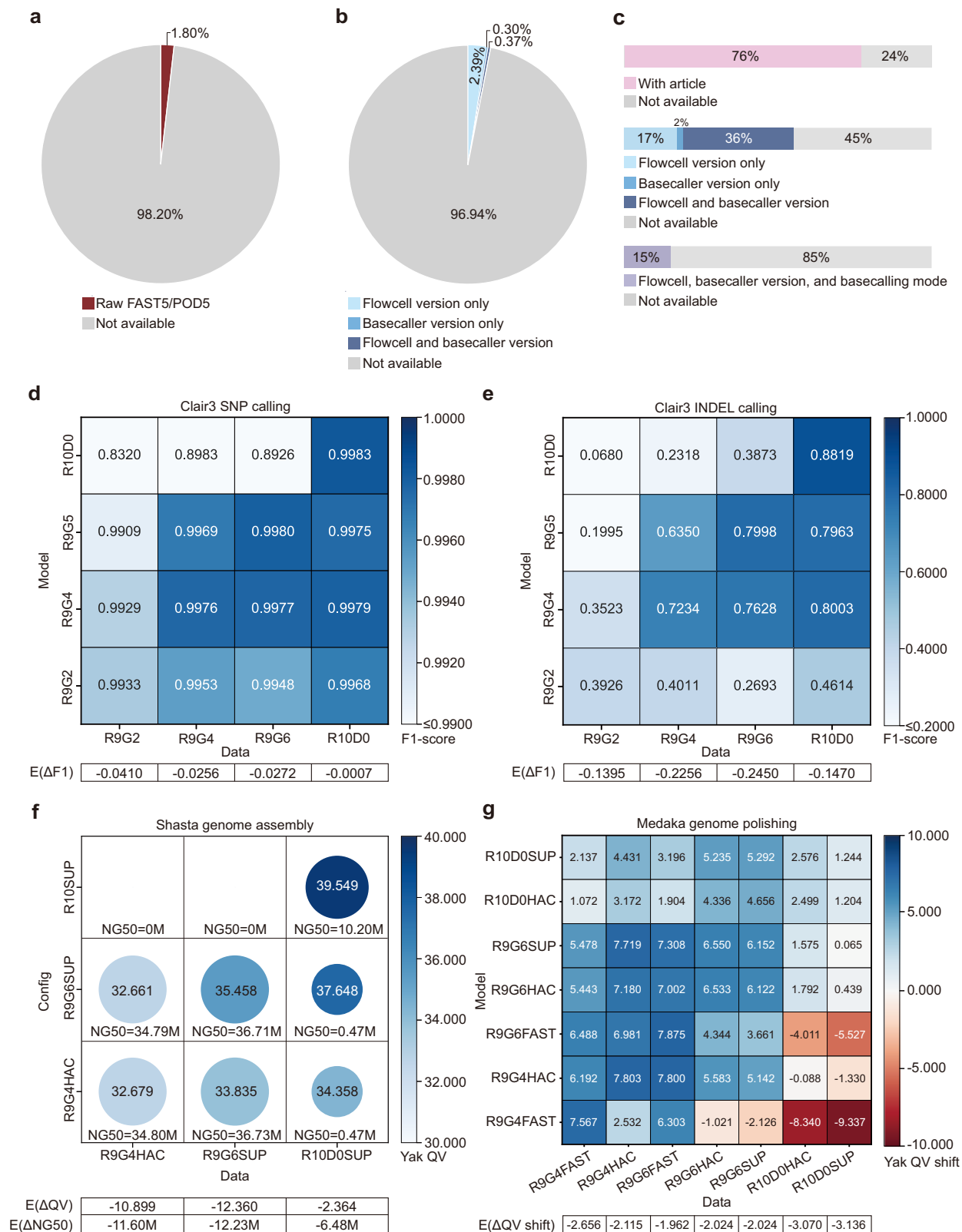
To tackle these challenges, we introduce a computational approach named LongBow (Lucid Dorado and Guppy Basecaller configuration predictor), which can predict flowcell types and basecaller configurations by analyzing the base quality value (QV) patterns in FASTQ files of ONT data. Extensively validated across 514 ONT FASTQ files from 66 datasets with 15 different combinations of flowcell types and basecaller configurations, LongBow achieves an accuracy of

95.33%. Leveraging LongBow, we have also created LongBowDB, a database hosting predicted metadata for all human ONT datasets in the SRA, accessible via SRA ID queries. We further demonstrate the utility of LongBow by reanalyzing the high-confidence datasets of COVID-19 Genomics UK (COG-UK) project, which hosts the ONT sequencing data of over 100,000 SARS-CoV-2 genomes but lacks information about flowcell types and basecaller configurations. The results show that the consensus sequences reported by COG-UK project can only be reproduced by reanalyzing the FASTQ files using the LongBow-predicted flowcell types and basecaller configurations. Moreover, we propose a variant calling pipeline, Artex, to reanalyze the COG-UK ONT data. The results show that Artex can significantly increase the accuracy of variant calling using the LongBow-predicted flowcell types and basecaller configurations. Notably, some large deletions, which might have a significant functional impact, are missed in all the samples by original consensus sequences published by COG-UK. These deletions can be correctly detected by Artex, which leverages LongBow-predicted flowcell types and basecaller configurations. These findings highlight the substantial potential of LongBow-based reanalysis of ONT data for generating valuable biological insights. By addressing the metadata gap, LongBow and LongBowDB not only enable more accurate and efficient analyses of public ONT data but also improve the reproducibility of ONT-related research, thereby facilitating discoveries through large-scale data integration.

Results

Incorrect flowcell type or basecaller configuration leads to substantially decreased performance of data analysis algorithms

Numerous state-of-the-art algorithms for ONT data analysis such as sequence alignment, variant calling, haplotype phasing, genome



assembly and genome polishing have a direct or indirect reliance on the specific flowcell type and basecaller configuration (Table 1). This dependency arises from the notable influence of these configurations on error rates, error patterns, and the distribution of base quality values (QV). Furthermore, it is important to highlight that the latest version of minimap2⁹, a widely utilized long reads aligner, has incorporated distinct parameters tailored for both the latest R10 and legacy

R9 ONT data. This will potentially influence a wide range of alignment-based analysis tools.

To investigate the consequences of using incorrect flowcell type and basecaller configuration, we evaluated three representative algorithms: Clair3¹⁰ (version 1.0.4) for variant calling, Shasta¹¹ (version 0.11.1) for genome assembly, and Medaka (version 1.11.3, <https://github.com/nanoporetech/medaka>) for genome polishing, utilizing

Fig. 1 | The significance of flowcell types and basecaller configurations in ONT data analysis. **a** Proportion of ONT data with raw FAST5/POD5 files in the SRA database. **b** Proportion of ONT data with specific flowcell types or basecaller configurations in the metadata of SRA. **c** Proportion of ONT data with specific flowcell types or basecaller configurations in the publications associated with 100 randomly selected SRA records. **d** Performance of Clair3 for SNP calling using correct and incorrect flowcell types or basecaller configurations. The color scale indicates the F1-score. The x-axis represents the flowcell type and basecaller configuration used for basecalling the test data. The y-axis represents the parameter configuration or pretrained model of the evaluated algorithm. Clair3 uses the same model for Guppy 3 and Guppy 4, and another shared model for Guppy 5 and Guppy 6. Flowcell type and basecaller configuration are encoded as strings, where the number after 'R' denotes the major flowcell version, 'G' represents Guppy, 'D' represents Dorado,

and the number following 'G' or 'D' is the major basecaller version. E(Δ F1) is the average F1-score loss when using random models. **e** Performance of Clair3 for INDEL calling. The definitions of x-axis and y-axis are identical to subfigure **d**. **f** Performance of Shasta for genome assembly. The definitions of x-axis and y-axis are similar to those in subfigure **d**, with an additional suffix to represent the basecalling mode. The color scale indicates Yak QV of the assembled contigs and the circle size indicates the decimal logarithm of contig NG50. M means million bases. E(Δ QV) and E(Δ NG50) are the average Yak QV loss and NG50 loss when using random parameters. **g** Performance of Medaka for genome polishing. The definitions of x-axis and y-axis are identical to those in subfigure **f**. The color scale indicates Yak QV shift of the assembled contigs after polishing. The encoding of flowcell type and basecaller configuration is identical to that in subfigure **f**. E(Δ QV shift) is the average Δ QV shift loss when using random parameters.

the ONT sequencing data of HG002¹², a widely used human sample for algorithms benchmarking (Methods). The optimal parameters for each algorithm, as stated in their respective documentation, are directly dependent on the specific flowcell types and basecaller configurations. Basecalling procedures were performed using Guppy versions 2.3.7, 4.2.2, and 6.3.8 for the R9.4.1 data, and Dorado version 0.4.3 for the R10.4.1 data. Guppy versions 3 and 5 were not included because these versions yield nearly identical basecalling results with versions 4 and 6, respectively, measured by sequence and QV similarity (Supplementary Fig. 2, Methods). Subsequently, the basecalled reads were aligned to the reference genome GRCh38 utilizing minimap2 (version 2.26-r1175).

We employed Clair3¹⁰ to call variants in the HG002 dataset using several pretrained models for different basecaller configurations: Guppy 2, Guppy 3/4, Guppy 5/6, and Dorado 0 (Supplementary Data 3). According to Clair3 documentation, the same pretrained models were used for both Guppy versions 3 and 4, as well as for Guppy versions 5 and 6. Furthermore, Clair3 utilizes a unified model for both the HAC and SUP modes of Guppy, with no provision for the FAST mode. The accuracy of the variant calling was assessed using hap.py (version 0.3.8-17-gf15de4a, <https://github.com/illumina/hap.py>) with the benchmark variants from Genome in a Bottle (GIAB)¹³. Our results indicate that optimal performance is contingent on selecting the pretrained model that aligns with the specific flowcell type and basecaller configuration of the sequencing data for both SNP and INDEL calling in most cases (Fig. 1d–e and Supplementary Fig. 3a–b). The only exception is that the R10 Dorado0 HAC model achieved slightly better accuracy than the R10 Dorado0 SUP model on the R10 Dorado0 SUP data for SNP calling (Supplementary Fig. 3a). The inappropriate application of pretrained models can result in significant declines in accuracy. For instance, using a pretrained R10 model on R9 Guppy4 data (where the data is sequenced by R9 flowcell and basecalled with Guppy major version 4), or a pretrained R9 Guppy2 model on R9 Guppy6 data led to a 0.49 and 0.53 reduction in the F1-score¹⁴ for INDEL calling respectively (Fig. 1e). The average F1-score loss ranges from 0.14 to 0.25 if the model is selected randomly. While the influence of flowcell type and basecaller configuration is more pronounced in INDEL detection than in SNP detection, using pretrained R10 model on the R9 Guppy2 data still leads to a 0.16 reduction in the F1-score in SNP detection (Fig. 1d). Although using an incorrect model does not cause a significant F1-score drop for R10 Dorado 0 data, in other cases, the average F1-score loss ranges from 0.03 to 0.04 if the model is selected randomly.

We conducted a comprehensive assessment of de novo genome assembly performance using Shasta¹¹, employing configurations tailored to R9 Guppy4 HAC, R9 Guppy6 SUP, and R10 SUP data (Supplementary Data 4). We evaluated assembly completeness and accuracy by calculating NG50¹⁵ and assembly consensus quality value (assembly QV) using calN50 (<https://github.com/lh3/calN50>) and Yak¹⁶ (version 0.1-r56, <https://github.com/lh3/yak>) respectively. Our results reveal that optimal assembly completeness and accuracy are achieved when utilizing configurations matched to the specific dataset (Fig. 1f and Supplementary Fig. 3c). Misapplication of configurations may lead

to assembly failures or significant declines in contig length or accuracy. For instance, applying Shasta's R10 SUP model to R9 Guppy4 HAC or R9 Guppy6 SUP data resulted in very short contigs which cannot cover even half of the reference genome (NG50 = 0). Applying Shasta R9 Guppy4 HAC or R9 Guppy6 SUP config to R10 SUP data substantially reduced the NG50 by 95.4% and yak QV by 13.1% and 4.81%, respectively (Fig. 1f). On average, randomly selecting parameters for Shasta leads to an NG50 drop ranging from 6.48 M to 11.60 M and a yak QV drop ranging from 2.364 to 12.360.

We additionally evaluated the efficacy of genome polishing employing the Medaka toolkit (<https://github.com/nanoporetech/medaka>). Notably, Medaka's architecture is customized to different flowcell types, basecallers, and basecalling modes. Consequently, we explored Medaka configurations tailored to seven unique sets of basecalled data, denoted as follows: R9 Guppy4 FAST, R9 Guppy4 HAC, R9 Guppy6 FAST, R9 Guppy6 HAC, R9 Guppy6 SUP, R10 Dorado0 HAC, and R10 Dorado0 SUP (Supplementary Data 5). The polishing efficacy was assessed by the shift in assembly QV between the initial Flye-assembled¹⁷ contigs and the Medaka-polished contigs. Our findings underscore that the optimal polishing outcomes are attainable mainly through the utilization of correct configurations except the R10 Dorado0 HAC and R9 Guppy6 HAC data, as illustrated in Fig. 1g. The Medaka parameters for R10 Dorado0 SUP data and R9 Guppy6 SUP achieved slightly better but comparable accuracies when applied to the R10 Dorado0 HAC data and R9 Guppy6 HAC data.

It is noteworthy that the computational methods we reviewed (Table 1), except Medaka, do not offer a pretrained model or parameter settings specifically for the FAST mode of Guppy 4, Guppy 6, or Dorado 0 (Guppy 2 lacks the option for FAST mode). Utilizing a pretrained model that is incompatible with the data basecalled by the FAST mode results in reduced performance. For example, applying Clair3 with R9 Guppy6 HAC/SUP model to the R9 data basecalled with Guppy6 FAST mode leads to a 0.46 reduction in F1-score compared to the results obtained from the same data basecalled using HAC mode for INDEL calling (Supplementary Fig. 3b). This decrease in performance may be attributed to either the reduced basecalling accuracy in FAST mode or the mismatch between the data and the pretrained model. Consequently, it is crucial to identify whether data were basecalled using FAST mode before proceeding with downstream analysis, to ensure the expected accuracy can be achieved.

Base QV pattern is an effective feature to distinguish flowcell types and basecaller configurations

LongBow predicts flowcell types and basecaller configurations by directly analyzing FASTQ files. This method circumvents the computation-intensive alignment process, relying solely on base quality values (QVs) for making predictions. We demonstrate that a straightforward yet effective method of feature extraction from base QVs is sufficient to differentiate between various flowcell types and basecaller configurations.

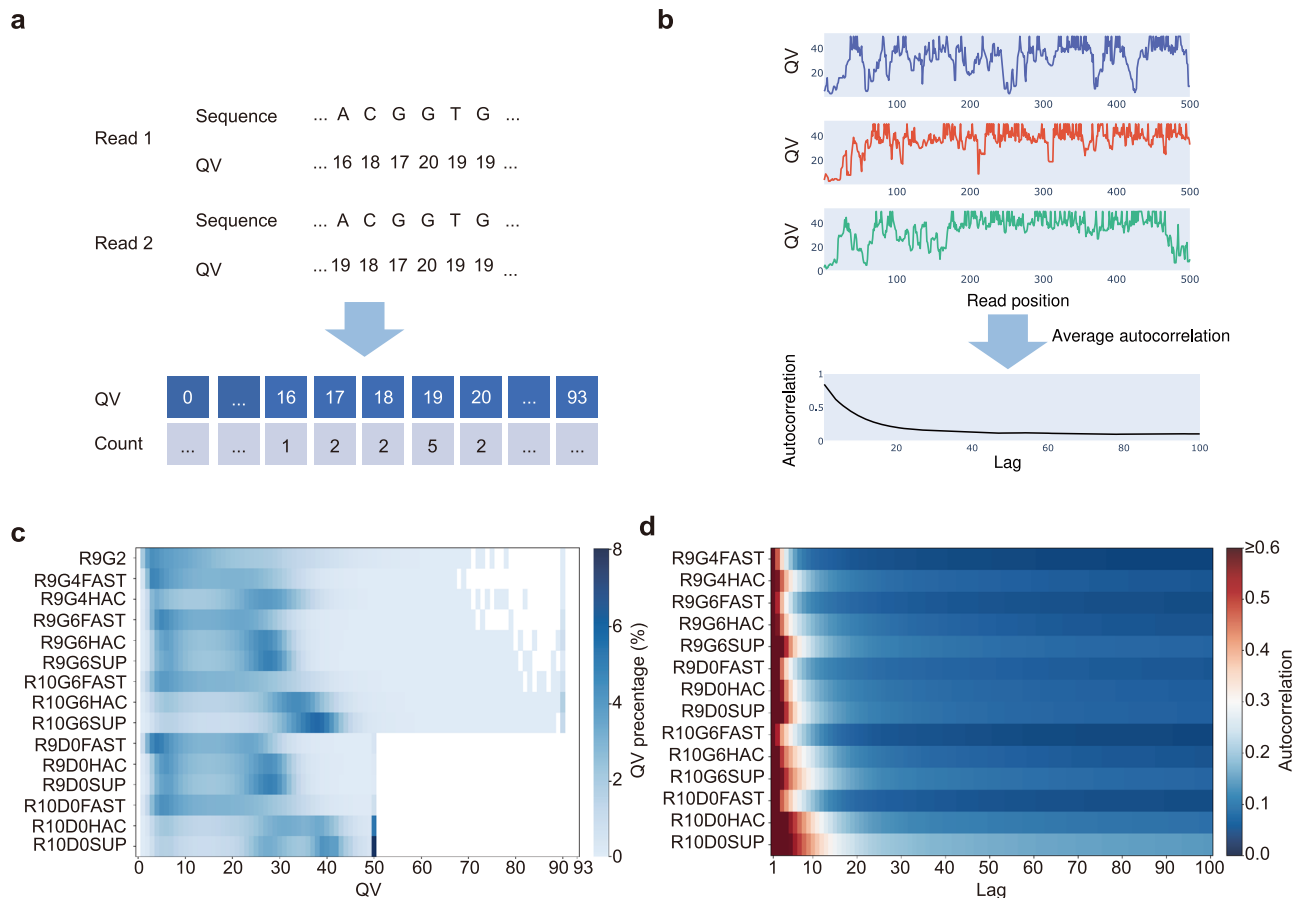


Fig. 2 | QV patterns of different flowcell types and basecaller configurations.
a Workflow for calculating QV distribution. **b** Workflow for calculating average QV autocorrelation. **c** Average QV distribution for each combination of flowcell type and basecaller configuration. The x-axis represents QV, while the y-axis represents flowcell type and basecaller configuration, encoded consistently with

Fig. 1d, Average autocorrelation for each combination of flowcell type and basecaller configuration. The definitions of the axes and color scale are identical to those in subfigure **c**. The encoding of flowcell type and basecaller configuration is consistent with Fig. 1.

Base QVs of ONT data follow the Phred+33 standard¹⁸, ranging from 0 to 93. We utilized two primary features to characterize the QV pattern. The first is the distribution of base QVs across a sequencing dataset. This distribution is estimated by counting the frequency of each QV from 0 to 93 (Fig. 2a). The second feature is the average autocorrelation between neighboring bases within a sequencing dataset (Fig. 2b).

For the training data, we collected 90 FASTQ files from 12 independent sequencing experiments covering seven model organisms, encompassing 15 different combinations of flowcell types and basecaller configurations (see Supplementary Data 6 for details, with additional information provided in the Methods section). Analysis of the training data reveals that average QV distributions vary visibly among different flowcell types and basecaller versions (Fig. 2c). Notably, the QV range in the Dorado basecaller is restricted from 1 to 50, in contrast to the 1 to 90 range observed in the Guppy basecaller (Fig. 2c). This stark contrast serves as a clear indicator for distinguishing between these two basecallers. Furthermore, within each basecaller, noticeable differences exist between various flowcell types and basecaller versions (Fig. 2c, Fig. 3a-b). Our findings also indicate that newer flowcells, basecaller versions, or more accurate basecalling modes generally produce higher base QVs, evidenced by a shift in the distribution toward higher values (Fig. 2c). While the QV distribution of the FAST mode of each basecalling configuration is distinct from that of the HAC or SUP mode, distinguishing between HAC and SUP modes based

solely on QV distribution is more difficult (Fig. 2c, Fig. 3a-b). These observations are supported by results from Multidimensional Scaling (MDS), which visualizes QV distributions from all FASTQ files in the training dataset (Fig. 3a-b).

To further differentiate between different basecalling modes within the same basecaller, we analyzed the base QV autocorrelation for each FASTQ file. The autocorrelation for each read was averaged to derive this feature (see details in Methods). Both the average QV autocorrelation of each combination of flowcell type and basecaller configuration (Fig. 2d) and MDS visualization (Fig. 3c-d and Supplementary Fig. 4) confirm that QV autocorrelation is an effective discriminative feature between the FAST, HAC and SUP modes. Results on Guppy 2 are omitted due to the absence of basecalling mode option in Guppy 2 (Supplementary Data 1).

Framework for predicting flowcell types and basecaller configurations

LongBow employs a hierarchical classification framework based on QV distribution and autocorrelation as depicted in Fig. 4a. In the first layer, LongBow determines the basecaller, either Guppy or Dorado, by examining if the maximum QVs is equal to 50. Specifically, as we have reported in the previous section, Dorado imposes a QV limit at 50, while Guppy's QVs can reach up to 90 (Fig. 2c). Subsequently, the second layer employs a K-Nearest Neighbors (KNN) classifier that utilizes QV distribution to predict the flowcell type and basecaller version for each branch from the first layer (Fig. 4a). The final layer of LongBow

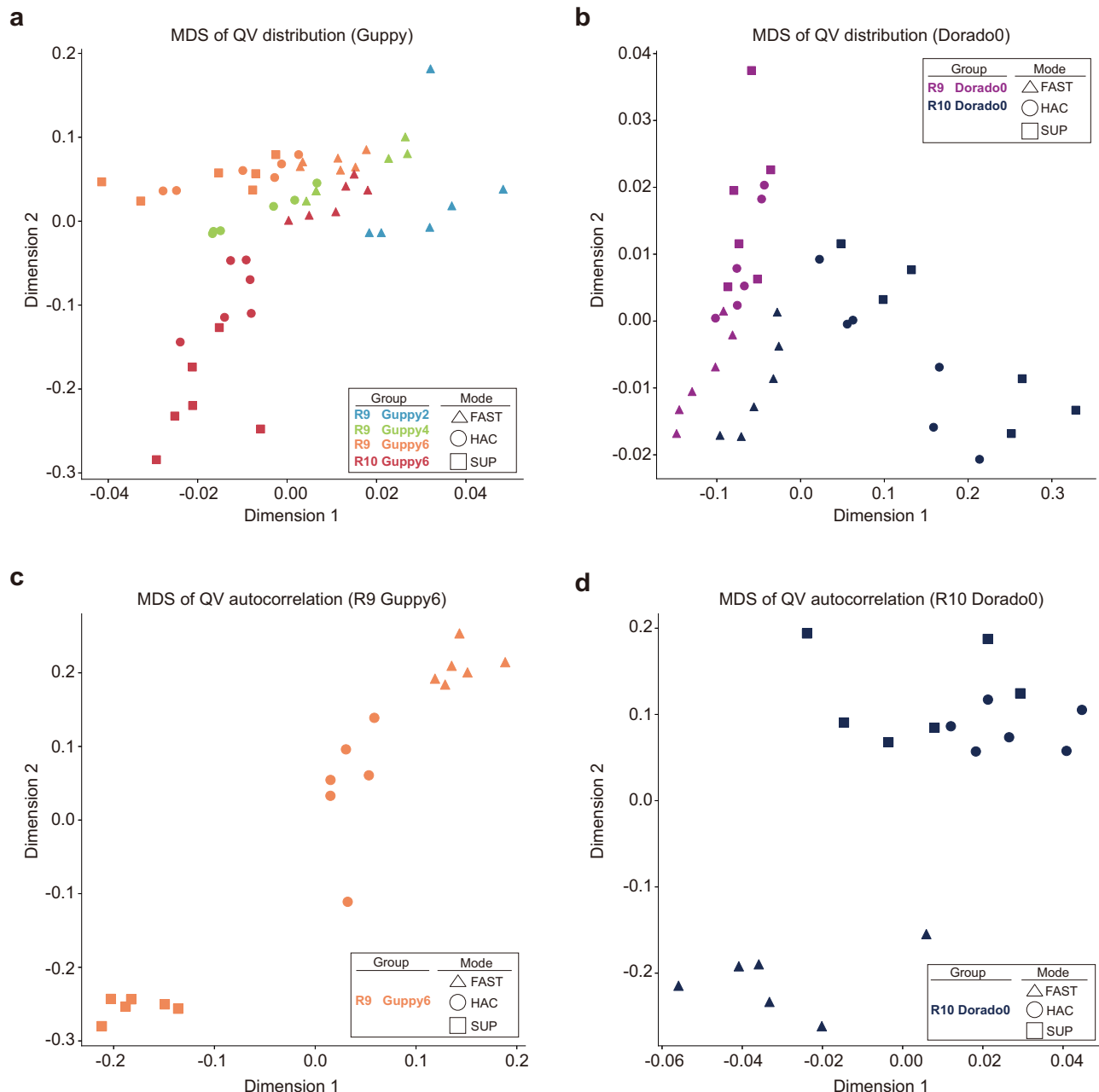


Fig. 3 | Multidimensional scaling of QV distribution and autocorrelation. Similarity between QV distributions in subfigure **a** and **b** is measured by Bhattacharyya distance. Similarity between QV autocorrelations in subfigure **c** and **d** is measured by Euclidean distance. Flowcell type and basecaller version are

represented by different colors and basecalling mode is represented by shape in multidimensional scaling (MDS). **a** MDS of QV distributions of Guppy. **b** MDS of QV distributions of Dorado. **c** MDS of QV autocorrelations of R9 Guppy6. **d** MDS of QV autocorrelations of R10 Dorado0.

applies another KNN classifier to distinguish between the basecalling modes based on QV autocorrelation (Fig. 4a). Notably, only Guppy version 3 and higher, as well as Dorado, support basecalling mode configuration (Supplementary Data 1).

It is worth noting that LongBow does not have to employ all the reads of a dataset to make predictions. Using a subset of the reads can substantially increase the speed of LongBow. We performed downsampling analysis using a sample from the Oxford Nanopore Open Data Project (s3://ont-open-data/colo829_2023.04/COLO829). The QV distributions and autocorrelations obtained using 10,000 reads show no obvious difference compared to those obtained using the entire dataset (Supplementary Fig. 5). We used 10,000 reads for each sample to evaluate LongBow's performance in this study.

Evaluating the performance of LongBow on independent testing data

We conducted a comprehensive evaluation of LongBow's accuracy and runtime performance across 66 independent ONT sequencing experiments, involving 44 species. Of these, 41 are model organisms selected based on previous research¹⁹, while the remaining 3 were chosen for their unique availability of cpDNA and viral cDNA (Fig. 4b and Supplementary Data 7). The raw FASTQ/PODS data of these experiments were basecalled to 514 FASTQ files with 15 distinct combinations of flowcell types and basecaller configurations. Basecalling was carried out using Guppy versions 2.3.7, 4.5.4 (FAST and HAC mode), Guppy 6.4.6 (FAST, HAC, and SUP mode), and Dorado 0.4.1 (FAST, HAC, and SUP mode). LongBow achieved an overall accuracy of

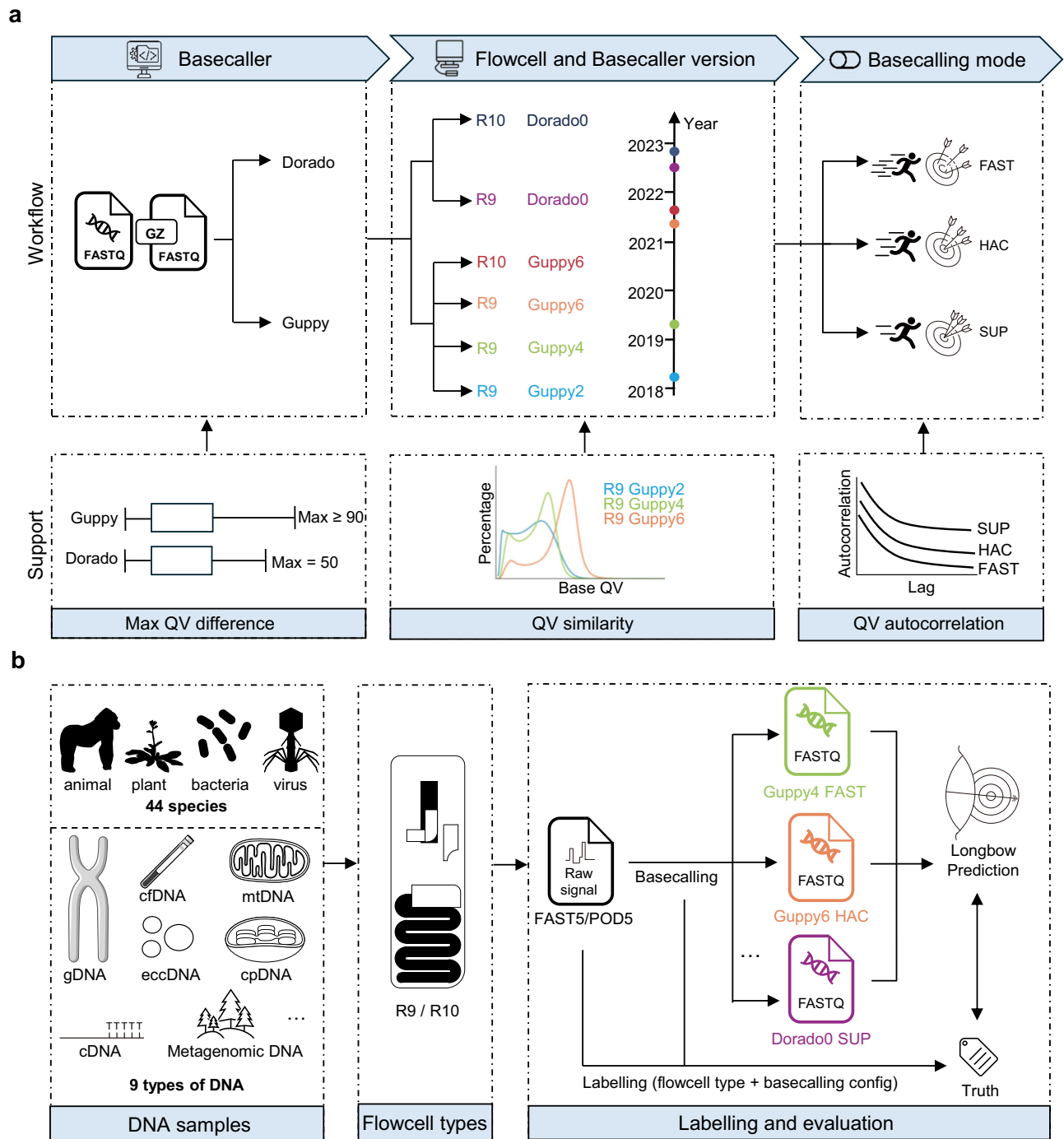


Fig. 4 | Workflow of LongBow. a The hierarchical framework of LongBow. The three layers of LongBow are presented at the top of the figure. The boxes at the bottom indicate the features used by each layer. The first layer classifies basecallers based on the maximum QV value of each sample. The second layer predicts flowcell type (R10 or R9) and basecaller version based on the Bhattacharyya distance of the QV distribution. The third layer predicts whether the basecalling mode is FAST, HAC, or SUP based on QV autocorrelation. **b** The pipeline for LongBow evaluation. Basecalling was performed to convert the raw signals in FAST5/POD5 files into basecalled FASTQ files. During this process, the flowcell types and basecaller configurations were extracted and recorded as truth. LongBow then predicted the flowcell types and basecaller configurations based on the FASTQ files. These predictions were compared with the truth to evaluate LongBow's accuracy. Three

species icons were obtained from PhyloPic (<https://www.phylopic.org>) *Gorilla gorilla* by Margot Michaud and *Arabidopsis thaliana* var. *thaliana* by Jake Warner are under Creative Commons Zero 1.0 Public Domain license (<https://creativecommons.org/publicdomain/zero/1.0/>). *Myoviridae* by Ninjatacoshell is under Creative Commons CC BY 4.0 (<https://creativecommons.org/licenses/by/4.0/>). Icons for gDNA, cfDNA, and mtDNA were sourced from Servier Medical Art (<https://smart.servier.com>) under Creative Commons CC BY 4.0 (<https://creativecommons.org/licenses/by/4.0/>). The cpDNA icon was obtained from Openclipart (<https://openclipart.org>), which is chloroplast by Torisan under Creative Commons Zero 1.0 Public Domain license (<https://creativecommons.org/publicdomain/zero/1.0/>).

95.33% (Fig. 5a), while demonstrating a runtime of approximately three seconds per sample (comprising 10,000 reads after downsampling) using 32 threads on a workstation equipped with dual Intel Xeon Platinum 8352Y CPUs, 512 GB of DDR4 memory, and an HDD array.

Since the downstream analysis algorithms require varying levels of detail regarding flowcell types and basecaller configurations (Table 1), we evaluated the accuracy of LongBow accordingly. Misclassification of HAC and SUP modes has a minimal impact on the accuracy of some downstream analysis algorithms such as Clair3, we reassessed LongBow's accuracy by merging HAC and SUP modes as the same class. This adjustment resulted in an accuracy of 96.89% (Fig. 5b). LongBow's accuracy is 97.67% for predicting the combinations of flowcell types and basecaller versions (Fig. 5c), and reaches 98.25% for solely predicting flowcell types (Fig. 5d). If only basecaller version or basecaller type is concerned, the accuracy is 99.22% (Fig. 5e) and 100.00% (Fig. 5f), respectively. Different accuracies should be referred to according to different scenarios of downstream analysis.

LongBow's versatility across an extensive range of species and sample preparation methods is particularly noteworthy, given the diversity of species and sequencing libraries represented in the test data. It is compatible with a broad array of organisms, including animals, plants, fungi, bacteria, and viruses. LongBow was also comprehensively tested on diverse sequencing library construction methods, such as standard whole-genome DNA sequencing, targeted sequencing of specific genomic regions such as ribosomal DNA, organelle DNA sequencing (e.g., mitochondrial or chloroplast DNA), sequencing of reverse-transcribed viral RNA, transcriptome analysis via cDNA sequencing, as well as ultra-short sequencing applications like cell-free DNA sequencing and the detection of cancer-related extra-chromosomal circular DNA (Supplementary Data 7). This broad versatility enables LongBow to handle large-scale data with robustness.

LongBowDB: a database to query the flowcell types and basecaller configurations for public ONT data

We applied LongBow to all ONT human DNA sequencing data in SRA (9643 runs including cDNA data as of January 9, 2024). The results show that the current public human ONT data in SRA are dominated by the data generated with R9 flowcell and basecalled with Guppy 5/6 or Guppy 3/4, composing a total of 89.82% (Fig. 6a). We compared LongBow-predicted flowcell types and basecaller configurations with those found in metadata or associated publications (1989 out of 9643). The results show that the overall consistency is 91.45% (Fig. 6b). LongBow demonstrated a consistency ranging from 95.22% to 100.00% across different levels of detail in flowcell types and basecaller configurations (Supplementary Fig. 6), further affirming its accuracy in recovering these metadata from public data.

To facilitate data mining and enhance community accessibility, we created LongBowDB, a database designed to store the LongBow-predicted flowcell types and basecaller configurations for all human ONT datasets in the SRA. Users can retrieve pre-computed flowcell types and basecaller configurations by searching for SRA IDs, as illustrated in Fig. 6c. The back-end data of LongBowDB is available in Supplementary Data 8.

LongBow improves the reproducibility of studies using ONT data

As discussed in previous sections, missing flowcell types or basecaller configurations in ONT data are common in both databases and publications (Fig. 1b-c). When a publication fails to report this critical metadata while relying on it for analysis, it creates reproducibility issues. For instance, the COVID-19 Genomics UK (COG-UK) project²⁰ has reported over 100,000 ONT sequencing datasets of SARS-CoV-2, but did not release flowcell types or basecaller configurations of these data in associated publications, SRA/ENA metadata, text description in the databases, or the associated BAM files (the BAM files are just

aligned reads rather than the basecaller-reported BAM files). The project provides consensus sequences reported by Artic (<https://artic.readthedocs.io/en/latest>), an experimental and computational pipeline that wraps Nanopolish²¹ and Medaka as its core engines for calling variants and generating consensus sequences. These consensus sequences are the starting point and key data source for multiple studies of SARS-CoV-2, such as studying the impact of variants on transmissibility²², large-scale community surveillance²³, and epidemiology²⁴⁻²⁶. Since Nanopolish requires raw signal files (FAST5) as input (<https://github.com/jts/nanopolish>), which are not released by the COG-UK project, the only way to reproduce the reported mutations or consensus sequences is by applying Artic with Medaka as the core engine. Medaka is explicitly dependent on the flowcell type and basecaller configuration and produces suboptimal results if using the incorrect parameters, as we reported (Fig. 1g).

We attempted to reproduce the reported consensus sequences using Artic with Medaka models based on LongBow-predicted, random selection, and default setting (details are in Methods) respectively on 269 independent ONT datasets with corresponding NGS (short-read Next Generation Sequencing) data in COG-UK (EMBL-EBI accession number is PRJEB37886, the list of RUN IDs is in Supplementary Data 9). The results demonstrate that using random or default Medaka models leads to a substantial drop in the accuracy of the consensus sequences. We benchmarked the accuracy of variant calling using the consensus sequences derived from NGS data as the gold standard. The ONT consensus sequence downloaded from EMBL-EBI has high consistency with the NGS-derived consensus sequences (F1-score is 0.9885 for SNP detection and 0.9858 for INDEL detection, Fig. 7a-b). In contrast, the consensus sequences reported by Artic using random or default Medaka models exhibit F1-score drops ranging from 0.0558 to 0.0874 for SNP detection and from 0.2826 to 0.8137 for INDEL detection compared to the ONT consensus sequences from EMBL-EBI (Fig. 7a-b). Reanalysis of the ONT data using Artic with Medaka models based on LongBow-predicted flowcell types and basecaller configurations has similar accuracy as the ONT consensus sequence downloaded from EMBL-EBI (F1-score is 0.9910 for SNP detection and 0.9846 for INDEL detection, Fig. 7a-b). These results underscore the importance of flowcell types and basecaller configurations in ONT data analysis, highlighting that LongBow is essential for ensuring reproducibility in ONT-based publications when this metadata is absent.

LongBow-based reanalysis of published ONT data offers valuable biological insights

Since ONT data analysis is challenging and the algorithms evolve rapidly, reanalysis of the published ONT raw FASTQ data with updated algorithms may yield previously unrecognized biological insights. In this context, we proposed an improved version of the Artic ONT data analysis pipeline called Artex (Artic extension) by integrating the results of Medaka and Clair3, which was not included in the original Artic pipeline. In Artex pipeline, Clair3 was used to rescue some low-quality variants filtered by the original Artic pipeline to achieve a better accuracy (Supplementary Fig. 7, details are in Methods).

We applied Artex to the COG-UK data described in the previous section using Medaka and Clair3 models based on LongBow-predicted flowcell types and basecaller configurations. The results show that the accuracy is improved for both SNP and INDEL detection (Fig. 7a-b). Importantly, we found that the 57-bp large deletion (g.27406_27462del, p. L5_C23 del) in ORF7a, detected by Artex, is completely missed by both the original Artic pipeline and the consensus sequences reported in the EMBL-EBI database (Fig. 7c-d). Moreover, this missing deletion is not a rare occurrence: five out of the 269 samples contain this deletion based on NGS data, but none of these were detected by the original Artic pipeline. In contrast, Artex successfully identifies all of them. While the functional impact has not been fully studied, this large deletion overlaps with the N-terminal

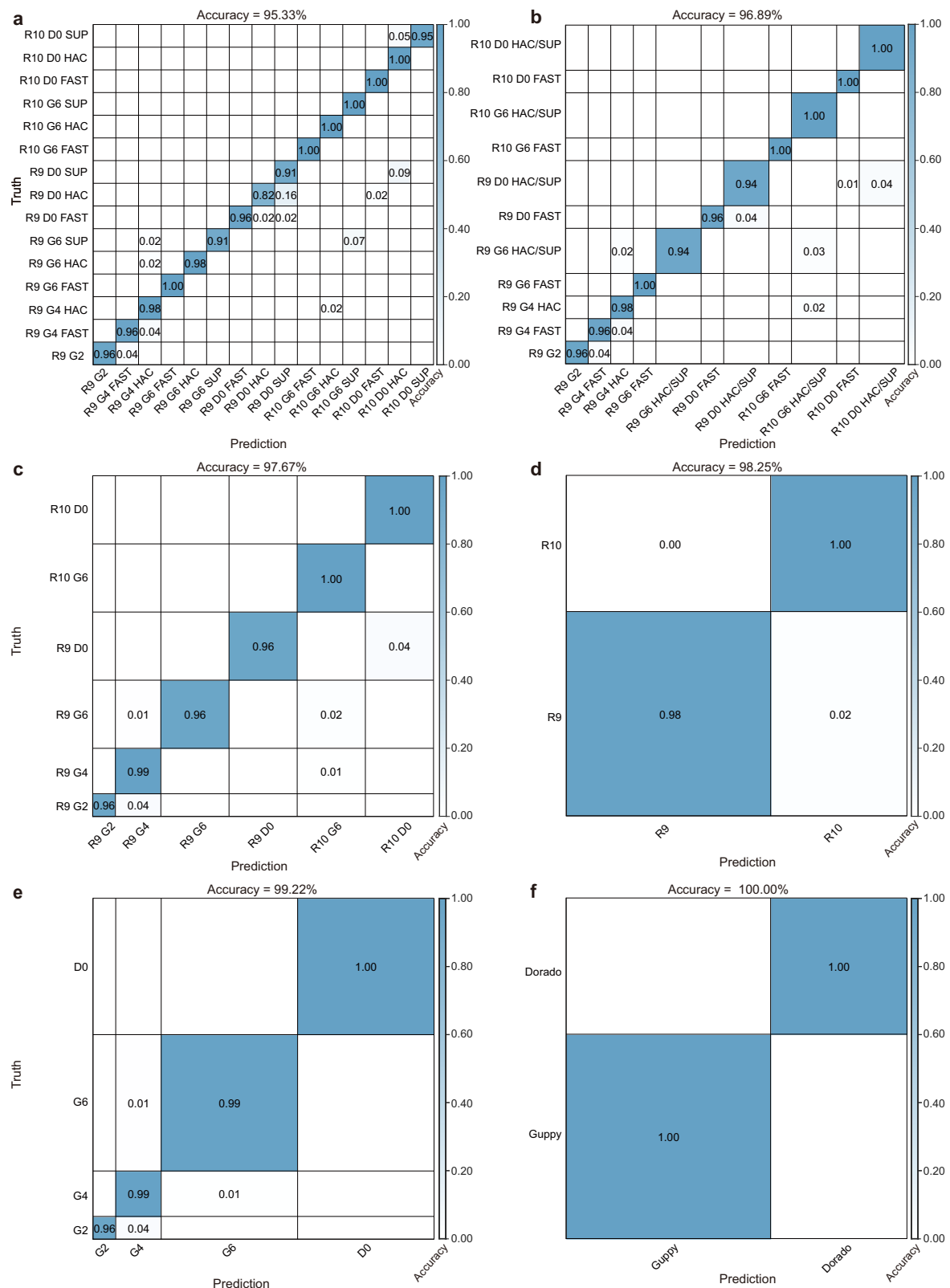


Fig. 5 | Accuracy of LongBow on independent test data. The x-axis represents the LongBow prediction, while the y-axis represents the true flowcell type and basecaller configuration. The encoding of flowcell type and basecaller configuration is consistent with Fig. 1. The color scale indicates the accuracy of each category. The number in each cell represents the proportion of LongBow predictions given the truth, with the summation of the numbers for each row equaling 1. **a** Accuracy of

predicting flowcell types and basecaller configurations. **b** Accuracy of predicting flowcell types and basecaller configurations by merging HAC and SUP modes. **c** Accuracy of predicting flowcell types and basecaller versions. **d** Accuracy of predicting flowcell types. **e** Accuracy of predicting basecaller versions. **f** Accuracy of predicting basecallers.

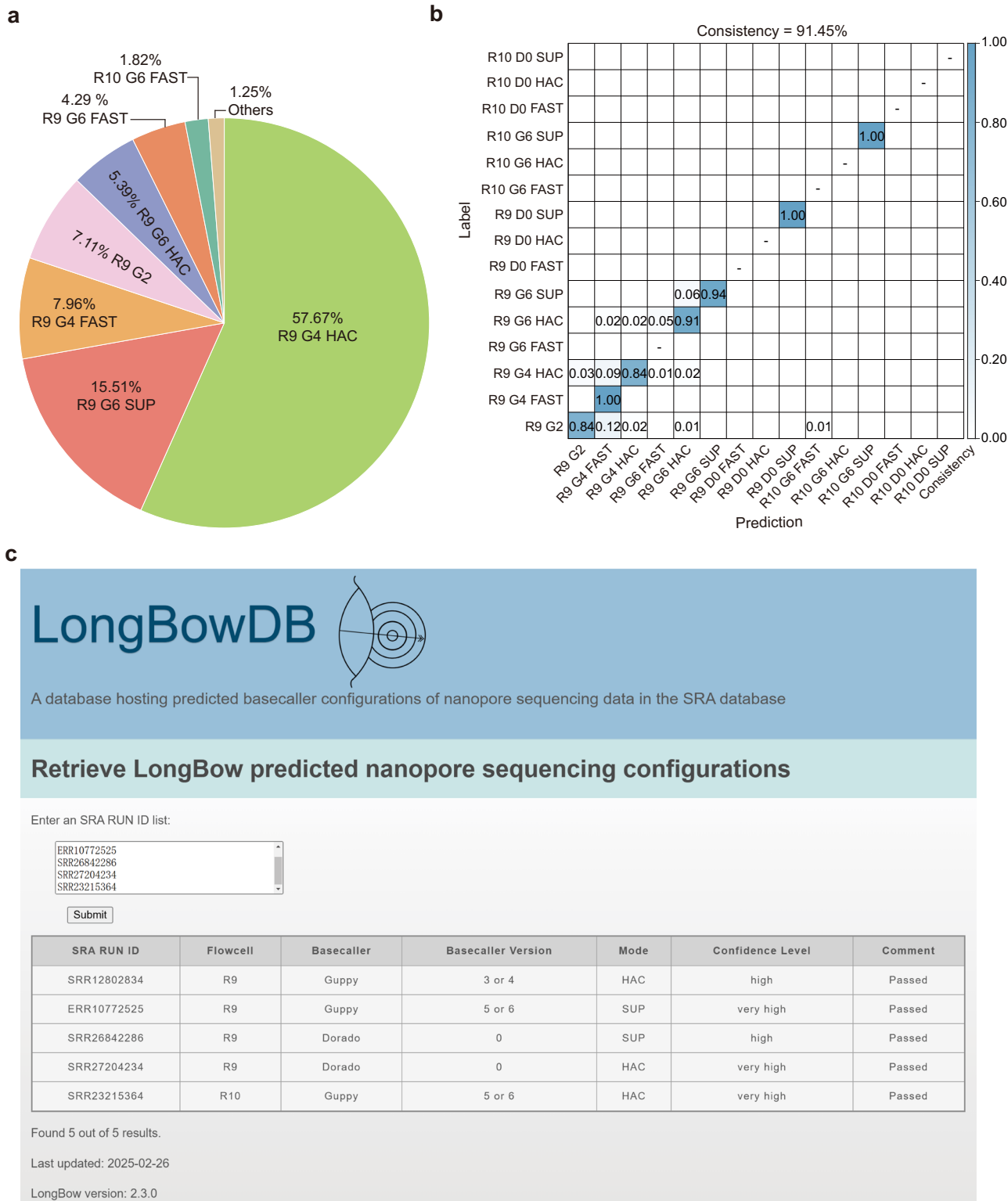
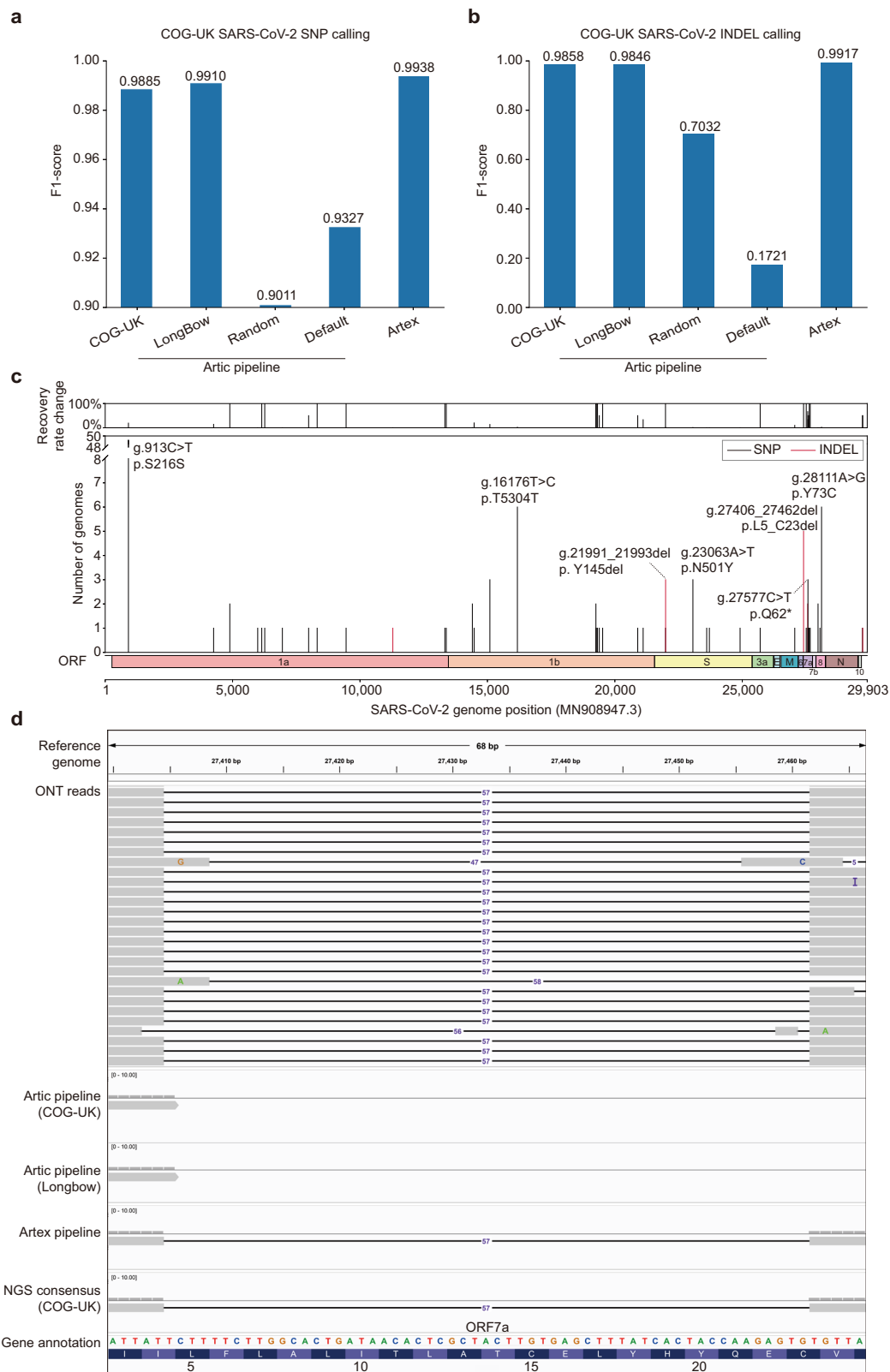


Fig. 6 | Applying LongBow to human ONT sequencing data. **a** Distribution of predicted flowcell types and basecaller configurations for all human ONT data in SRA up to January 9, 2024. **b** Consistency between LongBow predictions and flowcell type and basecaller configuration found in the metadata or associated publications. The x-axis represents the LongBow prediction, while the y-axis “label” represents the flowcell type and basecalling configuration found in the metadata or

associated publications. The encoding of flowcell type and basecaller configuration is consistent with Fig. 1. The color scale indicates the consistency of each category. The number in each cell represents the proportion of LongBow predictions given the label, with the summation of the numbers for each row equaling one. **c** Snapshot of LongBowDB, the web server to query flowcell types and basecaller configurations based on LongBow predictions.



signal peptide (residues 1-15) of ORF7a, which might influence its localization²⁷.

We further investigated the function of the additional variants detected by Artex, compared to those identified in the original Artex pipeline consensus sequences. The results show that Artex detected 121 additional correct variants compared to Artex (Supplementary Data 10). A variant detected by Artex or Artex is considered correct if it

is also identified by NGS. Across the 121 additional variants, the increased recovery rate of these variants ranges from 0.39% to 100% (Fig. 7c). For each variant, the recovery rate is defined as the ratio of variants detected by both ONT and NGS to those detected by NGS. Some of these extra variants have been reported as hallmarks of enhanced infection and transmission, or are essential for the escape from neutralizing antibodies. For example, the N501Y mutation

Fig. 7 | LongBow-based reanalysis of the ONT datasets from COG-UK project. **a** SNP calling performance. The x-axis represents different methods: “COG-UK” refers to variant calling based on consensus sequences downloaded from the EMBL-EBI database; “LongBow” refers to the Artic pipeline using LongBow-predicted Medaka models; “Random” refers to the Artic pipeline using randomly chosen Medaka models; “Default” refers to the Artic pipeline using the default Medaka model (r1041_e82_400bps_sup_v4.3.0); “Artex” refers to the Artex pipeline using LongBow-predicted Medaka and Clair3 models. The y-axis represents the F1-score for SNP calling. **b** INDEL calling performance. The x-axis and y-axis are identical to those in subfigure **a**. **c** Additional variants detected by the Artex pipeline compared to the original Artic pipeline. In the bottom panel, the open reading frames (ORF) are represented by different colors. The middle panel represents the number of additional variants detected by Artex. The top panel represents the recovery rate improvement for each of them. All three panels share the same x-axis,

which represents the SARS-CoV-2 genome position in reference genome MN908947.3. **d** Snapshot of g.27406_27462del detected by Artex in the COG-UK sample ALDP-12BA880 (ENA accession ERR5399848), visualized in the Integrative Genomics Viewer (IGV). Each track, from bottom to top, represents different data: “Gene annotation” shows the gene annotations for the SARS-CoV-2 genome (MN908947.3); “NGS Consensus (COG-UK)” shows the NGS-based consensus sequence from the COG-UK project; “Artex Pipeline” shows the consensus sequence reported by Artex; “Artic Pipeline (LongBow)” shows the consensus sequence reported by the Artic pipeline using LongBow-predicted Medaka models; “Artic Pipeline (COG-UK)” shows the ONT-based consensus sequence from the COG-UK project; “ONT reads” shows the aligned ONT sequencing data; and “Reference Genome” shows the genomic position of the SARS-CoV-2 reference genome (MN908947.3).

(g.23063 C > T) in the spike protein has been shown to increase the affinity of the viral spike protein for cellular receptors, thereby enhancing viral transmission²⁸. Additionally, the Y145del mutation (g.21991_21993del) has been frequently observed in viral variants that escape neutralization by NTD antibodies targeting the spike protein²⁹. A full list of the functional impacts of these variants is provided in Supplementary Data 10. Given their functional importance, failure to detect these variants could have significant implications for both clinical practice and research.

In addition to variant analysis, we also conducted lineage analysis of the consensus sequences. Using the consensus lineage assignments from NGS and ONT as benchmarks (see Methods for details), we found that the consensus sequence produced by the Artex pipeline has better accuracy (1.11% increase) in lineage assignment compared to the consensus sequence downloaded from the EMBL-EBI database using the Artic pipeline (Supplementary Data 9). Although the increase in accuracy is modest, we argue that even a 1% error rate in lineage assignment can result in a significant number of incorrectly assigned lineages in large-scale epidemiological studies, potentially leading to over 1,000 errors in a project sequencing more than 100,000 samples. While this example focuses on SARS-CoV-2, the same strategy using LongBow can be applied to any public ONT data to reanalyze the raw FASTQ files, potentially yielding biological insights through updated algorithms or by exploring various aspects of the data.

Discussion

The integration of large-scale public ONT sequencing data holds immense potential for uncovering biological insights, such as characterizing genomic variants at a population level, identifying pathogenic variants, and exploring evolutionary dynamics across multiple species. However, realizing these opportunities currently faces significant challenges due to the absence of essential metadata, namely flowcell type and basecaller configuration, which are pivotal for state-of-the-art analysis methods. This critical gap not only limits comprehensive data mining and downstream analysis of existing ONT datasets but also leads to serious reproducibility issues in ONT-related research.

To fill this gap, we introduce LongBow, a computational method designed to infer flowcell type and basecaller configuration from base QV patterns extracted from FASTQ files of ONT sequencing data. Leveraging extensive testing, LongBow demonstrates both high speed and high accuracy in restoring these vital parameters. Additionally, to enhance accessibility, we developed LongBowDB, a database with a user-friendly interface, enabling researchers to query flowcell types and basecaller configurations for all human ONT sequencing data in SRA. As a use case, we applied LongBow to the COG-UK dataset, a large-scale sequencing project of SARS-CoV-2. Our results show that LongBow is crucial for accurately reproducing the reported genomic variants. We then developed a LongBow-based variant caller that can identify significantly

more validated and functionally important variants, providing valuable biological insights.

In this study, we focused on data from R9/R10 flowcells and Guppy/Dorado basecallers. While deprecated flowcell types, such as R7, and earlier basecallers, such as Albacore or third-party basecallers, have been utilized in previous research, we contend that the majority of publicly available ONT data are derived from R9/R10 flowcells and processed with Guppy/Dorado basecallers. This is due to the widespread adoption of R9/R10 as the dominant flowcell types in ONT sequencing, coupled with Dorado being the recommended basecaller for R10 data and Guppy being designated for R9 data. We also focused on simplex sequencing data and mainstream R9.4 and R10.4 flowcell, while deprecated duplex techniques, such as 1D² and 2D sequencing and experimental flowcell such as R9.5 and R10.3, in total account for only a small portion (0.55%) in the public database.

LongBow demonstrates robust performance in most circumstances, regardless of species or DNA sample type. However, its accuracy declines in cases involving target sequencing of highly repetitive regions or samples with artificially introduced modifications, such as GpC modification in NanoNOME. Analysis of LongBow's error patterns, based on the data in Supplementary Data 8, reveals that 61.76% of errors originate from these samples. Most of these edge cases exhibit abnormally low QV and are misclassified by LongBow as older flowcell types, basecallers, or less accurate basecalling modes (100% for target sequencing of highly repetitive regions and 76.74% for samples with artificially introduced modifications). Users should be cautious when dealing with these samples. Not only is LongBow less accurate, but it is also unclear whether the “correct” pre-trained models or parameters of downstream analysis algorithms, such as variant calling, remain optimal, as these samples may exhibit different error patterns than those assumed by these algorithms.

Public databases hosting sequencing data, such as SRA, have made attempts to include flowcell type and basecaller configuration in their metadata. However, a significant proportion of this information remains absent, with approximately 96.94% of metadata fields lacking such details. Looking ahead, we propose that these databases or external projects consider integrating LongBow or LongBowDB directly into their infrastructure to enrich metadata. Additionally, we advocate that academic journals and databases mandate the inclusion of flowcell type and basecaller configuration for all submitted ONT data. A possible technical solution is to require the submission of basecalled BAM files instead of FASTQ files, as basecalled BAM file directly produced by Guppy or Dorado produced typically contain details about flowcell type and basecaller configuration in the BAM header. Despite the rich information in BAM files, they are often not uploaded due to challenges such as larger file sizes, accessibility issues, and limited database support. As a result, the vast majority of ONT data in databases like SRA are available only as database-generated FASTQ files, which do not include information about flowcell types or basecaller configurations.

As sequencing technology continues to advance, we anticipate that future ONT data analysis algorithms will increasingly account for flowcell types and basecaller configurations. For instance, the recent update of the ONT basecaller Dorado introduced a transformer-based basecalling algorithm in SUP mode, claimed to achieve Q28 basecalling accuracy on R10.4 5 kHz data. Given the significant reduction in sequencing error rates, downstream analysis methods may need to update their parameters or models to maintain optimal performance. As the new basecaller models mature, LongBow will also be updated accordingly to incorporate these advancements.

Methods

Obtaining statistics of ONT records in SRA database

To calculate the proportion of records consisting of raw FAST5/PODS files, we navigated to the advanced search section of the [SRA](#). In the “Builder” section, we selected “oxford nanopore” under “Platform” and set the “Publication Date” from “2010/01/01” to “2024/01/09” to obtain the total number of ONT data in SRA, denoted as N_{total} . Similarly, we selected “oxford nanopore” under “Platform”, set the “Publication Date” from “2010/01/01” to “2024/01/09”, and included “filetype nanopore” under “Properties” to obtain the number of ONT SRA Runs with raw FAST5/PODS files, denoted as N_{raw} . The proportion of records consisting of raw FAST5/PODS files was then calculated by N_{raw}/N_{total} .

To calculate the proportion of records consisting of BAM/CRAM file, we selected “oxford nanopore” under “Platform”, set the “Publication Date” from “2010/01/01” to “2024/01/09”, and included “filetype bam” or “filetype cram” under “Properties” to obtain the number of ONT SRA Runs with BAM/CRAM file, denoted as N_{bam} . The proportion of records consisting of BAM/CRAM files was then calculated by N_{bam}/N_{total} .

Given that SARS-CoV-2 accounts for a significant portion of the data, we also calculated this proportion by excluding it. We selected “oxford nanopore” under “Platform”, set the “Publication Date” from “2010/01/01” to “2024/01/09”, and excluded “Severe acute respiratory syndrome-related coronavirus” under “Organism” to obtain the total number of ONT SRA runs excluding SARS-CoV-2, denoted as E_{total} . We selected “oxford nanopore” under “Platform”, set the “Publication Date” from “2010/01/01” to “2024/01/09”, set “filetype bam” or “filetype cram” under “Properties” and excluded “Severe acute respiratory syndrome-related coronavirus” under “Organism” to obtain the number of ONT SRA Runs with BAM/CRAM file while excluding SARS-CoV-2, denoted as E_{bam} . The proportion of records consisting of raw BAM/CRAM files was then calculated by E_{bam}/E_{total} .

The number of records containing flowcell type and basecaller configuration in the metadata was obtained by exploring the complete metadata in XML format, which was downloaded from https://ftp.ncbi.nlm.nih.gov/sra/reports/Metadata/NCBI_SRA_Metadata_Full_20240120.tar.gz. An in-house Python script was developed to analyze the entire metadata set. Specifically, this script first filtered the total SRA Runs based on two conditions: (1) the publication date of the SRA Run falls within the range of “2010/01/01” to “2024/01/09”; (2) the sequencing platform is Oxford Nanopore. Multiple regular expressions were then employed to identify keywords related to flowcell or basecaller configuration information within the XML files of the filtered SRA Runs. Finally, the script summarized and output the proportion of SRA Runs that included flowcell type or basecaller configuration data.

Since it is prohibitively time-consuming to exhaustively search for flowcell type and basecaller configuration in all associated publications, we randomly selected 100 SRA Runs from the nanopore sequencing data in the SRA, excluding those related to severe acute respiratory syndrome coronavirus. We then conducted comprehensive manual literature mining for these 100 selected SRA Runs as well as for all human ONT data, covering 9643 SRA Runs from 275 Bio-Projects, as of January 9, 2024. We followed these steps: 1) search for sequencing kit and flowcell type in FAST5/PODS file if available; 2)

search for flowcell type and basecalling configurations in the BAM file headers if available; 3) extract metadata that may provide relevant information, including project title, publication ID or Digital Object Identifier (DOI), and keywords; 4) search for the publication if a PubMed ID or DOI is provided; 5) search for the project title in Google Scholar if a PubMed ID or DOI is not provided; 6) If no associated publication is found, search the authors’ publication records and link the data to relevant publications based on the authors’ names, institutions, publication time, location, and manually curated keywords in the metadata; 7) scan the publication for details on library construction, flowcell types, sequencing kit, basecalling software, basecaller version, basecalling mode, and configuration files. When discrepancies arise between the information presented in the metadata and that found in the associated publication, we use the information provided in the metadata as the definitive label.

Basecalling of the benchmark HG002 dataset

We downloaded the raw FAST5 files of HG002 for R9.4.1 data and R10.4.1 data from s3://ont-open-data/gm24385_2020.11/ and s3://ont-open-data/giab_lsk114_2022.12 respectively. Basecalling was performed with Guppy 2.3.7, Guppy 4.2.2, and Guppy 6.3.8 for the R9.4.1 data. Basecalling of the R10.4.1 data was performed by Dorado 0.4.3 (<https://github.com/nanoporetech/dorado>). For Guppy, we used the command “guppy_basecaller -r -i \${fast5/pod5} -s \${output} -c \${config.cfg} -x cuda:0,1”. For Dorado, we used the command “dorado_basecaller \${config} --emit-fastq -x cuda:0,1 -r \${fast5/pod5} > \${output.fastq}”. The basecalled reads were filtered by specific QV cutoffs according to each basecalling mode using Chopper³⁰ (version 0.7.0, <https://github.com/wdecoster/chopper>).

Sequence identity and QV score similarity between Guppy versions

For sequence identity and QV score similarity between different Guppy versions, we choose the widely used ONT sequencing dataset of HG002 sequenced with R9.4.1 flowcell (s3://ont-open-data/gm24385_2020.11/), for evaluation. Guppy 2.3.7, Guppy 3.6.1 (FAST, HAC), Guppy 4.5.4 (FAST, HAC), Guppy 5.1.16 (FAST, HAC, SUP), Guppy 6.4.6 (FAST, HAC, SUP) are used for basecalling. We merged all the FASTQ files from both the PASS and FAIL folders reported by Guppy.

We used mappy (Python api of minimap2, <https://github.com/lh3/minimap2/tree/master/python>) to conduct pairwise sequence alignment between the two reads with the same read ID basecalled from two different Guppy versions. Sequence identity was calculated using the length of the matching bases (mlen in mappy) divided by the length of the alignment (blen in mappy).

$$\text{Sequence identity} = \frac{\text{mlen}}{\text{blen}} \quad (1)$$

The QV score similarity was evaluated by Bhattacharyya coefficient using the ‘distances’ package (<https://pypi.org/project/distances/>) in Python. For probability distributions M and N in the same domain of definition D , the Bhattacharyya coefficient (BC) between distribution M and N , is defined as follow. The Bhattacharyya distance (BD) used in this article is the negative logarithm of the Bhattacharyya coefficient.

$$\text{BC}(M, N) = \sum_{x \in D} \sqrt{M(x)N(x)} \quad (2)$$

$$\text{BD}(M, N) = -\ln(\text{BC}(M, N)) \quad (3)$$

Benchmarking the variant calling algorithm Clair3 with different pretrained models

The basecalled FASTQ files were aligned to GRCh38 reference genome (GenBank assembly ID GCA_000001405.15) by minimap2 (version 2.26-r1175) with the command “minimap2 -t 128 -L -o {output} -ax map-ont {refseq} {input.fastq}”. The minimap2-aligned BAM files were used as input for variant calling with Clair3 using different pretrained models listed in Supplementary Data 3. Pretrained Clair3 models were downloaded from http://www.bio8.cs.hku.hk/clair3/clair3_models and https://github.com/nanoporetech/terio/tree/master/clair3_models. We used the command “run_clair3.sh --bam_fn={bamfile} --ref_fn={refseq} --threads=40 --platform=ont --model_path={model} --output={output}” to detect variants with Clair3. The detailed Clair3 models are listed in Supplementary Data 3. The resulting VCF file from Clair3 was compared with the benchmark variants set (https://ftp-trace.ncbi.nlm.nih.gov/ReferenceSamples/giab/release/AshkenazimTrio/HG002_NA24385_son/latest/GRCh38/HG002_GRCh38_1_22_v4.2.1_benchmark.vcf.gz) using hap.py (version v0.3.8-17-gf15de4a, <https://github.com/illumina/hap.py>) with the command “hap.py {truth.vcf} {clair3.vcf} -o {output} -r {refseq} -f {confident_region.bed}”. The “confident_region.bed” file was obtained from https://ftp-trace.ncbi.nlm.nih.gov/ReferenceSamples/giab/release/AshkenazimTrio/HG002_NA24385_son/latest/GRCh38/HG002_GRCh38_1_22_v4.2.1_benchmark_noinconsistent.bed.

Benchmarking the de novo genome assembly algorithm Shasta with different configurations

The filtered reads were used as input for de novo genome assembly in Shasta (version 0.11.1, <https://github.com/paoloshasta/shasta>). We tested three different Shasta configurations: Nanopore-Sep2020 (for R9G4HAC data), Nanopore-May2022 (for R9G6SUP data), and Nanopore-R10-Fast-Nov2022 (for R10SUP data), according to the documentation of Shasta, using the command “shasta-Linux-0.11.1 --input {input.fastq} --config {shasta_config} --threads 128 --assemblyDirectory {output}”. The detailed configuration is listed in Supplementary Data 4.

The NG50 of the resulting contigs was calculated using calN50 (<https://github.com/lh3/calN50>) with the command “calN50.js -f {refseq.fasta.fai} {assembly.fasta}”. We also evaluated the assembly QV score using Yak (version 0.1-r56, <https://github.com/lh3/yak>). Briefly, we used Illumina 30X PCR-free short reads (https://s3-us-west-2.amazonaws.com/human-pangenomics/index.html?prefix=NHGRI_UCSC_panel/HG002/hpp_HG002_NA24385_son_v1/ILMN/downsampled/) data to build the yak k-mer table and used this table to evaluate the assembly QV for the Shasta assembly with the commands “yak count -b 37 -t 32 -o {ngs.yak} <(cat {R1.fq}) <(cat {R2.fq})” and “yak qv -t 32 -p -K 3.2g -l 100k {ngs.yak} {assembly.fasta} > {output.txt}”.

Benchmarking the genome polishing algorithm Medaka with different models

To evaluate the genome polishing performance of Medaka (version 1.11.3, <https://github.com/nanoporetech/medaka>) under different models, we chose seven different basecalling configurations for HG002 data listed in Supplementary Data 5. We did not include the R10 Dorado0 FAST data for testing because of the absence of corresponding Medaka model. Flye¹⁷ (version 2.9.3-b1797, <https://github.com/mikolmogorov/flye>) was used to assemble a draft for each group because Flye is the recommended draft assembler for Medaka (<https://github.com/nanoporetech/medaka>). The configuration of Flye is listed in Supplementary Data 5. The unpolished assembly output of Flye was used as input for Medaka polishing.

We tested seven different Medaka models listed in Supplementary Data 5 and also evaluated the impact of incorrect models. Each unpolished input assembly was evaluated with Yak to calculate the

draft assembly QV score (QV_{draft}) as the baseline. We used the command “medaka_consensus -i {basecalled_pass.fastq} -d {contigs.fastq} -o {output_dir} -m {medaka_model} -t 48 -b 50” to run assembly polishing. The polished assembly with the correct or incorrect Medaka models were again evaluated by Yak to calculate the polished assembly QV (QV_{polished}). We calculated the QV shift (ΔQV) to evaluate the performance of the genome polishing process using the following equation.

$$\Delta QV = QV_{\text{polished}} - QV_{\text{draft}} \quad (4)$$

QV definitions and calculations

In this section, several QV terms are introduced and explained to prevent any potential confusion. Base QV (QV_{base}), also known as the Phred quality score, is generated by the basecaller. QV_{base} is directly related to the estimated error rate of the base (P_{error}). The definition of QV_{base} is as follows:

$$QV_{\text{base}} = -10 \log_{10} P_{\text{error}} \quad (5)$$

QV_{base} is represented as ASCII character in basecalled FASTQ files. FASTQ files for ONT sequencing follow the Phred+33 standard. Following this standard, we retrieved QV_{base} by converting the ASCII character to integer using the built-in ‘ord’ function in Python and then subtracting the offset.

$$QV_{\text{base}} = \text{ord}(\text{ASCII}) - 33 \quad (6)$$

Read QV (QV_{read}) is the average of the per-base error rate of a sequence on the log scale, with a total of N bases in a given sequence. Read QV score is used for filtering low-quality reads. The definition of QV_{read} is as follows.

$$QV_{\text{read}} = -10 \log_{10} \left[\frac{1}{N} \sum 10^{-\frac{QV_{\text{base}}}{10}} \right] \quad (7)$$

Assembly QV reported by Yak (version 0.1-r56, <https://github.com/lh3/yak>) is the estimated average base accuracy of the assembly. According to its documentation, Yak used an empirical model to assess the assembly error rate by comparing assembly sequences to the k-mer spectrum of short reads. The assembly yak QV score was calculated using the estimated error rate of the assembly.

Extracting QV patterns from FASTQ files

We extracted QV distribution as described in Fig. 2a. Specifically, we denote the QVs of the i^{th} read as $R_i = \{q_{i1}, q_{i2}, \dots, q_{im_i}\}$, and calculated the frequency of each QV value ranging from 0 to 93 according to the formula $QVF = \left\{ \frac{\sum_{i=1}^n \sum_{j=1}^{m_i} I(q_{ij} = t)}{\sum_{i=1}^n m_i} \mid t = 0, 1, \dots, 93 \right\}$. As depicted in Fig. 2b, the autocorrelation with lag k of each read is calculated according to

$$ACF_i(k) = \frac{\sum_{j=1}^{m_i-k} (q_{ij} - \bar{q}_i)(q_{i(j+k)} - \bar{q}_i)}{\sum_{j=1}^{m_i} (q_{ij} - \bar{q}_i)^2} \quad (8)$$

and weighted average autocorrelation with lag k of a dataset is defined as

$$ACF(k) = \frac{\sum_{i=1}^n (m_i - k) ACF_i(k)}{\sum_{i=1}^n (m_i - k)} \quad (9)$$

The autocorrelation of a dataset is $ACF = \{ACF(k) | k = 1, 2, \dots, p\}$, where p is the maximal lag. Reads with length shorter than the maximal lag is omitted from autocorrelation calculation. The autocorrelation of each read is implemented by using the 'acf' function in Python package 'statsmodels' (<https://pypi.org/project/statsmodels>).

Preparation of the training dataset for LongBow

We collected publicly available ONT R9 data, including raw FAST5 files, for six common model organisms: *Homo sapiens*, *Mus musculus*, *Drosophila melanogaster*, *Arabidopsis thaliana*, *Saccharomyces cerevisiae* and *Escherichia coli* (Supplementary Data 6). Due to the relative scarcity of R10 data compared to R9 data, we were unable to find public ONT R10 data with raw FAST5 or POD5 files for *Mus musculus* and *Saccharomyces cerevisiae* at the time of model training. As an alternative, we used R10 data for the following five model organisms: *Homo sapiens*, *Danio rerio*, *Drosophila melanogaster*, *Arabidopsis thaliana*, and *Escherichia coli* (Supplementary Data 6), with two R10 datasets for *Homo sapiens*. In total we used 6 independent groups of ONT R9 data and 6 independent groups of ONT R10 data for LongBow model training.

For basecalling the R9 data, we employed Guppy versions 2.3.7 and 4.5.4 with both FAST and HAC modes, as well as Guppy version 6.4.6 and Dorado version 0.4.1 with FAST, HAC, and SUP modes. We chose the highest available sub-version of each basecaller to represent the final developed state of each main version. Given that Guppy versions 2.3.7 and 4.5.4 do not support R10.4 data, we used Guppy version 6.4.6 and Dorado version 0.4.1 for basecalling the R10 data, utilizing FAST, HAC, and SUP modes. Each combination of flowcell type, basecaller, basecaller version, and specific basecalling mode was defined as a label, resulting in a total of 15 classes, with 9 classes for R9 data and 6 classes for R10 data. The QV pattern was defined as the feature of the proposed machine learning model. The basecalling commands were similar to those provided in the section Basecalling of the benchmark HG002 dataset.

Hierarchical classification model of LongBow

The top layer of the model classifies basecaller based on whether the QV distribution is capped at 50 (Fig. 4a). If the QV distribution is capped at 50, the basecaller is classified as Dorado; otherwise, it is classified as Guppy. Formally, the basecaller is classified according to the following formula.

$$\text{Basecaller}(QVF) = \begin{cases} \text{Dorado}, & \sum_{i=51}^{93} QVF_i = 0 \text{ and } QVF_{50} \neq 0 \\ \text{Guppy}, & \sum_{i=51}^{93} QVF_i \neq 0 \text{ or } QVF_{50} = 0 \end{cases} \quad (10)$$

QVF is the QV distribution defined in section Extracting QV patterns from FASTQ files. The second layer consists of two separate K-Nearest Neighbors (KNN) classifiers for Guppy and Dorado configurations, respectively. The KNN classifier for Guppy uses QV distribution as input and includes nine labels: R9G2, R9G4FAST, R9G4HAC, R9G6FAST, R9G6HAC, R9G6SUP, R10G6FAST, R10G6HAC, and R10G6SUP. The KNN classifier for Dorado also accepts QV distribution as input and includes six labels: R9D0FAST, R9D0HAC, R9D0SUP, R10D0FAST, R10D0HAC, and R10D0SUP. The labels are encoded as strings, where the number following 'R' denotes the major flowcell version, 'G' represents Guppy, 'D' represents Dorado, the number following 'G' or 'D' is the major basecaller version, and the suffix denotes the basecalling mode, if applicable. To measure the similarity between QV distributions of different samples for the KNN model in this layer, we used the Bhattacharyya distance³¹, which is designed for comparing two probability distributions. Bhattacharyya distance is calculated using the 'distances' package in Python (<https://pypi.org/project/distances/>). The parameter K for KNN was set to 3.

In the third layer, if a sample is classified as R9G2, we report this predicted label as the final prediction and the third layer is skipped. However, if a sample is classified as R9G4FAST, R9G4HAC, R9G6FAST, R9G6HAC, R9G6SUP, R9D0FAST, R9D0HAC, R9D0SUP, R10G6HAC, R10G6SUP, R9D0HAC, R9D0SUP, R10D0HAC, or R10D0SUP, the basecalling mode label will be re-predicted by the third layer. In the third layer, for the prediction of R9G6 and R10G6, read QV cutoff was used to determine the mode, which is Q8 for FAST, Q9 for HAC, and Q10 for SUP. If a sample matches these predefined cutoffs, the basecalling mode is directly output. The samples with unclassifiable QV cutoff were further determined by five autocorrelation-based KNN classifiers for R9G4, R9G6, R10G6, R9D0, and R10D0, respectively. Autocorrelation is defined in the section Extracting QV Patterns from FASTQ Files. Unlike the second layer, we used Euclidean distance to measure the similarity between the autocorrelation of different samples for the KNN models in this layer, with K set to 3. The optimal lags of autocorrelation, which are 10 for R9G4, 9 for R9G6, 2 for R9D0, 100 for R10G6, and 3 for R10D0, were determined by leave-one-out cross-validation test on the training data. To classify samples with different read QV cutoffs, we employed a self-adaptive strategy in the second and third layers, both of which utilize KNN-based classifiers. We constructed multiple KNN models for each read QV cutoff and selected the model that matched the cutoff of the input dataset.

We also added a distance-based confidence score to the predictions of KNN model. For each data point, if the predicted class is G , the distance-based confidence score is calculated using the following formula:

$$\text{confidence score} = \frac{\sum_{i \in G} \frac{1}{d_i}}{\sum_i \frac{1}{d_i}} \quad (11)$$

Where d_i is the distance between testing data point to the i th training data point. The confidence scores of the KNN models in the second and third layers are averaged to obtain the final reported score.

Evaluation of the efficacy of data downsampling

We selected a high-coverage ONT sequencing dataset of human COLO829 melanoma fibroblasts (ATCC CRL-1974) from the ONT Open Data project ([s3://ont-open-data/colo829_2023.04/COLO829](https://ont-open-data/colo829_2023.04/COLO829)) to evaluate the efficacy of downsampling. This dataset contains 13,760,388 reads from its original POD5 files. We used Guppy 6.4.6 FAST mode and Dorado 0.4.1 FAST mode to basecall the entire POD5 dataset. To minimize the impact of read filtering, the basecalled FASTQ files from both the PASS and FAIL folders were merged following Guppy basecalling.

Using the basecalled FASTQ file as input, we randomly downsampled it to 1,000,000 reads, 100,000 reads, 10,000 reads, 1000 reads, 100 reads, and 10 reads, respectively, using seqtk (version 1.4-r122, <https://github.com/lh3/seqtk>). The QV distribution and autocorrelation were calculated for each downsampled dataset, as well as for the entire dataset. The efficacy of data downsampling was evaluated based on the maximum distribution differences and maximum autocorrelation differences.

Preparing the independent test dataset

To make the test data more comprehensive and representative, we selected 66 groups of test data from 44 organisms. Forty model organisms were chosen based on a previous review of model organism evolution¹⁹. SARS-CoV-2, HIV, and Lambda virus were also included to test the compatibility of LongBow. *Narcissus pseudonarcissus* was selected due to the availability of chloroplast DNA sequencing data. Raw FAST5/POD5 files were retrieved from various projects or studies, contributing to the diversity of the test data. This dataset also represents a wide range of sequencing libraries, including standard whole-genome DNA sequencing, targeted sequencing of specific genomic

regions such as ribosomal DNA, organelle DNA sequencing (mitochondrial or chloroplast DNA), sequencing of reverse-transcribed viral RNA, cDNA sequencing, metagenome DNA sequencing and ultra-short sequencing applications such as cell-free DNA sequencing and extra-chromosomal circular DNA. Supplementary Data 7 provides details of the test dataset, including species names, flowcell types, sequencing kit, DNA types, download sources, file formats, sampling frequencies, and average read lengths.

The graphical illustration of the LongBow testing pipeline is shown in Fig. 4b. Since it is computationally expensive to basecall test datasets, the raw FAST5/POD5 files were first randomly downsampled to 10,000 reads with `ont fast5_api` (version 4.1.1 https://github.com/nanoporetech/ont_fast5_api) or `pod5` tools (version 0.2.4, <https://github.com/nanoporetech/pod5-file-format>). The samples sequenced with R9 flowcell were basecalled with R9G2, R9G4FAST, R9G4HAC, R9G6FAST, R9G6HAC, R9G6SUP, R9D0FAST, R9D0HAC, and R9D0SUP, respectively. The R10 data were basecalled with R10G6FAST, R10G6HAC, R10G6SUP, R10D0FAST, R10D0HAC, and R10D0SUP, respectively. Basecalling with Guppy 2 failed for 17 samples due to compatibility issues. After basecalling with different configurations, we constructed the true label for each FASTQ file in these independent test datasets. The basecalled FASTQ files were then subjected to LongBow for configuration prediction. The predicted labels were compared to the true labels to evaluate LongBow's performance on the independent test dataset. In total, 514 FASTQ files were used for LongBow performance evaluation.

Restoring flowcell types and basecaller configurations for human ONT data in the SRA database

We downloaded the first 10,000 reads of each SRA Run in FASTQ format from all human ONT data in SRA database publication date until January 9, 2024 (Supplementary Data 8) with the command “`fastq-dump ${sra_run_id} -N 1 -X 10000 -Q 33 -O ${output}`”. The tool, `fastq-dump`, is part of the SRA Toolkit (version 3.0.7) downloaded from <https://github.com/ncbi/sra-tools>. 8,639 out of 9,643 Runs (89.59%) were downloaded successfully. The remaining downloads failed due to either restricted access or absences of FASTQ files under specific SRA entries. LongBow was used to predict flowcell types and basecaller configurations using the following command: `longbow.py -i ${input.-fastq} -b -o ${output.json} -t 48`. The database, LongBowDB, was built based on LongBow prediction. LongBowDB website takes a list of SRA Run IDs as input and interactively queries for the result of LongBow predicted flowcell, basecaller, basecaller version, and basecalling mode. The database website was developed with HTML5 and hosted on GitHub.

Development and maintenance of LongBowDB

We developed a one-click program to create and update the backend data of LongBowDB. First, we used NCBI's API, Entrez Direct (v23.4), to query the SRA database for new runs of human ONT sequencing data using the command: “`esearch -db sra -query “Homo sapiens [Organism] AND Oxford Nanopore [Platform] AND dna data [Filter]” | efetch -format runinfo > {latest_runinfo}`”. Next, new runs were downloaded using `fastq-dump` (SRA Toolkit v3.0.7) with the command: “`fastq-dump {runid} -N 1 -X 10000 -O {target_dir}`”. LongBow was then used to predict the flowcell type and basecaller configuration of each run. The updates were added to LongBowDB, along with the date and LongBow version. These steps were automated through `crontab` jobs on our servers, ensuring regular updates. Finally, manual review was performed before publishing the results to GitHub.

Benchmarking LongBow's resource consumption

For hardware, we used a server running Ubuntu 22.04.1 with dual Intel Xeon Platinum 8352Y CPU, 512 GB of DDR4 memory, and an HDD array to conduct the benchmark. The thread number for LongBow

parallelization was set to 32. The runtime was measured using the built-in ‘time’ package in Python.

Preprocessing of COG-UK data

We downloaded the COG-UK metadata from EMBL-EBI under accession number PRJEB37886 and used a Python script to identify data with the same COG-UK ID sequenced with both short-read Illumina sequencing (NGS) and long-read Oxford nanopore sequencing (ONT). In total, 269 independent ONT datasets with corresponding NGS data in COG-UK were selected for the following analysis (The list of IDs is in Supplementary Data 9). The consensus sequences of these data also are available from EMBL-EBI. To extract genomic variants from these consensus sequences, we first used `minimap2` (version 2.28-r1209) to align the consensus sequences to the SARS-CoV-2 reference genome (MN908947.3). The aligned BAM files were then processed with `bcftools`³² (version 1.20) for pileup and variant calling, using the command: “`bcftools mpileup -B -m 1 -f ${reference.fasta} ${sorted.bam} -O b | bcftools call -O v -o ${out.vcf} -mv --ploidy 1`”. The variants called from the consensus sequences were used as the truth set.

Reproducing SARS-CoV-2 variants with the Artic pipeline

To reproduce the variants of SARS-CoV-2 samples, we used the Artic pipeline (version 1.2.4, <https://github.com/artic-network/fieldbioinformatics>) for variant calling with the command “`artic minion --medaka --medaka-model ${medaka_model} --threads 16 --scheme-directory ${primerscheme_directory} --read-file ${input.-fastq} nCoV-2019/V3 ${output_name} --no-longshot --skip-nanopolish`”. The primer scheme files and SARS-CoV-2 reference file were downloaded from <https://github.com/phac-nml/SARS-CoV-2-resources>.

To evaluate the reproducibility of COG-UK-reported variants, we tested three strategies for selecting the Medaka model: 1) the LongBow-predicted Medaka model, 2) a randomly chosen Medaka model, and 3) the default Medaka model. For randomly chosen Medaka model, we used Python package ‘random’ to generate a list of random seeds, which were then used to determine the random Medaka model. For default Medaka model, we used “r1041_e82_400bps_sup_v4.3.0”, the default model for Medaka version 1.11.3, as wrapped in the latest Artic pipeline.

After running the Artic pipeline for three different strategies, variants were called from the consensus sequences as stated in section Preprocessing of COG-UK data. The variants were compared to those identified by NGS using `hap.py` (version 0.3.8-17-gf15de4a, <https://github.com/Illumina/hap.py>) with command: “`hap.py ${query.vcf} ${NGS_truth.vcf} -o ${output} -r ${reference.fasta} --threads 12 --quiet --set-gt hom`”.

For each strategy for selecting the Medaka model, we calculated the overall F1-score by summing up the true positive (TP_i), false positive (FP_i), and false negative (FN_i) of the 269 samples reported by `hap.py`. The definition of precision, recall, and F1-score was defined as that of `hap.py` (<https://github.com/Illumina/hap.py/blob/master/doc/happy.md>).

$$\text{precision} = \frac{\sum_{i=1}^{269} TP_i}{\sum_{i=1}^{269} TP_i + \sum_{i=1}^{269} FP_i} \quad (12)$$

$$\text{recall} = \frac{\sum_{i=1}^{269} TP_i}{\sum_{i=1}^{269} TP_i + \sum_{i=1}^{269} FN_i} \quad (13)$$

$$F1 - \text{score} = \frac{2 \times \text{precision} \times \text{recall}}{\text{precision} + \text{recall}} \quad (14)$$

Reanalysis of COG-UK Data with the Artex pipeline

Multiplex PCR is at the core of the Artex experimental pipeline, used to amplify the SARS-CoV-2 genome. However, due to amplification bias and unbalanced template amounts caused by potential sample RNA degradation, some regions of the genome may be sequenced with relatively low coverage. In the original Artex pipeline, variants with low coverage were excluded from the final consensus sequence and reported as FAIL. We developed Artex pipeline on the basis of Artex pipeline (Supplementary Fig. 5). In addition to the original Artex pipeline, we used Clair3¹⁰ (version 1.0.10) for variant calling. Since some sites may have low coverage due to the aforementioned issues, we set the ‘min_coverage’ parameter of Clair3 to a relatively low value of 10. The Clair3-reported variants were then compared to the ‘FAIL’ variants reported by the Artex pipeline with bcftools (version 1.14) ‘isec’ command. The shared variants were then merged with the original ‘PASS’ variants with bcftools (version 1.14) ‘merge’ command. The consensus sequences of Artex pipeline were generated using bcftools (version 1.14) ‘consensus’ command. The subsequent variant evaluation follows the methodology outlined in section Reproducing SARS-CoV-2 Variants with the Artex pipeline.

Annotation and visualization of SARS-CoV-2 variants

Gene names, mutations, and mutation types of the 121 additional variants identified by the Artex pipeline were annotated using COV2-Var database³³ (<https://biomedbdc.wchscu.cn/COV2Var/>). The genomic mutation and protein mutation follows the nomenclature of HGVS standard (<https://hgvs-nomenclature.org/stable/recommendations/general/>). Protein annotations for each variant were obtained from the GFF3 file downloaded from the NCBI database (https://www.ncbi.nlm.nih.gov/datasets/genome/GCF_009858895.2/). The functions of SARS-CoV-2 proteins were labeled using the UniProt database³⁴.

The example variants were visualized using Integrative Genomics Viewer³⁵ (IGV, version 2.15.1) as in Fig. 7d. We used the SARS-CoV-2 Wuhan-Hu-1 isolate (MN908947.3) as reference genome in IGV. For the track of raw ONT reads, Artex pipeline (COG-UK), Artex pipeline (LongBow), Artex pipeline, and NGS consensus, we loaded the minimap2-aligned BAM file into IGV. For the gene annotation track, we loaded the GFF3 annotation file downloaded from NCBI database (https://www.ncbi.nlm.nih.gov/datasets/genome/GCF_009858895.2/) into IGV.

Determine the lineage of SARS-CoV-2 samples

To construct the lineage truth of 269 SARS-CoV-2 samples, we used the NGS consensus sequence in combination with the ONT consensus sequence downloaded from EMBL-EBI database. Pangolin³⁶ (version 4.3) was used to determine the lineage of all SARS-CoV-2 samples by the command “pangolin \${consensus.fasta} -o \${out_dir} --outfile \${sample_name}.csv -t 1” to predict the lineage of each pair of consensus sequences. In cases where the coverage of the consensus sequence fell below 0.7, we used the command “pangolin \${consensus.fastq} -o \${out_dir} --outfile \${sample_name}.csv -t 1 --max-ambig 1” to allow the lineage prediction to proceed. For the discordant results between NGS and ONT consensus sequences, we used Nextclade³⁷ (version 3.8.2, <https://clades.nextstrain.org/>) to conclusively determine the lineage of the ambiguous samples. We used Pangolin (version 4.3) to assign lineage for EMBL-EBI downloaded consensus sequences and Artex-assemble consensus sequences with the command “pangolin \${consensus} -o \${out_dir} --outfile \${sample_name}.csv -t 1”. The lineage assignment results were then compared to the forementioned truth.

Hardware and software implementation

All the basecaller information is summarized in Supplementary Data 1. To be noticed, Guppy 2 failed to run on Nvidia GPU with Ampere architecture (RTX 30 series) or Ada Lovelace architecture (RTX

40 series) due to compatibility issues, so Guppy 2 was run on dual Nvidia RTX 2070. For Guppy 4, Guppy 6, and Dorado 0, we used dual Nvidia RTX 3090 for basecalling. The Python scripts presented in this article were developed using Python 3.7.3. The conda environments were built with Conda 24.1.2 (Miniconda3). The MATLAB scripts were developed using MATLAB R2023a, License 31095115.

Reporting summary

Further information on research design is available in the Nature Portfolio Reporting Summary linked to this article.

Data availability

The SRA IDs or download links of the ONT sequencing data used in this work are listed in Supplementary Data 6, Supplementary Data 7, Supplementary Data 8, and Supplementary Data 9.

Code availability

The source code of LongBow is available at <https://github.com/JMencius/LongBow>. The source code of Artex is available at <https://github.com/JMencius/Artex>. The LongBowDB interactive website can be accessed at <https://jmencius.github.io/LongBowDB>. The source code of LongBowDB is available at <https://github.com/JMencius/LongBowDB>. Additionally, the in-house code and detailed documentation to reproduce the results are available at <https://github.com/JMencius/Arrow>. The source code of LongBow, Artex, LongBowDB can be found at Zenodo (<https://doi.org/10.5281/zenodo.15117614>)³⁸.

References

- Wang, Y., Zhao, Y., Bollas, A., Wang, Y. & Au, K. F. Nanopore sequencing technology, bioinformatics and applications. *Nat. Biotechnol.* **39**, 1348–1365 (2021).
- Cuber, P. et al. Comparing the accuracy and efficiency of third generation sequencing technologies, Oxford Nanopore Technologies, and Pacific Biosciences, for DNA barcode sequencing applications. *Ecol. Genet. Genom.* **28**, 100181 (2023).
- Shao, Y. et al. Phylogenomic analyses provide insights into primate evolution. *Science* **380**, 913–924 (2023).
- Gao, H. et al. The landscape of tolerated genetic variation in humans and primates. *Science* **380**, eabn8153 (2024).
- Beyter, D. et al. Long-read sequencing of 3,622 Icelanders provides insight into the role of structural variants in human diseases and other traits. *Nat. Genet.* **53**, 779–786 (2021).
- Wu, Z. et al. Structural variants in the Chinese population and their impact on phenotypes, diseases and population adaptation. *Nat. Commun.* **12**, 6501 (2021).
- Gamaarachchi, H. et al. Fast nanopore sequencing data analysis with SLOW5. *Nat. Biotechnol.* **40**, 1026–1029 (2022).
- Sanderson, N. D. et al. Comparison of R9.4.1/Kit10 and R10/Kit12 Oxford Nanopore flowcells and chemistries in bacterial genome reconstruction. *Micro. Genom.* **9**, 000910 (2023).
- Li, H. Minimap2: pairwise alignment for nucleotide sequences. *Bioinformatics* **34**, 3094–3100 (2018).
- Zheng, Z. et al. Symphonizing pileup and full-alignment for deep learning-based long-read variant calling. *Nat. Comput. Sci.* **2**, 797–803 (2022).
- Shafin, K. et al. Nanopore sequencing and the Shasta toolkit enable efficient de novo assembly of eleven human genomes. *Nat. Biotechnol.* **38**, 1044–1053 (2020).
- Zook, J. M. et al. Extensive sequencing of seven human genomes to characterize benchmark reference materials. *Sci. Data* **3**, 160025 (2016).
- Zook, J. M. et al. Integrating human sequence data sets provides a resource of benchmark SNP and indel genotype calls. *Nat. Biotechnol.* **32**, 246–251 (2014).

14. Hicks, S. A. et al. On evaluation metrics for medical applications of artificial intelligence. *Sci. Rep.* **12**, 5979 (2022).
15. Earl, D. et al. Assemblathon 1: A competitive assessment of de novo short read assembly methods. *Genome Res.* **21**, 2224–2241 (2011).
16. Cheng, H., Concepcion, G. T., Feng, X., Zhang, H. & Li, H. Haplotype-resolved de novo assembly using phased assembly graphs with hifiasm. *Nat. Methods* **18**, 170–175 (2021).
17. Kolmogorov, M., Yuan, J., Lin, Y. & Pevzner, P. A. Assembly of long, error-prone reads using repeat graphs. *Nat. Biotechnol.* **37**, 540–546 (2019).
18. Cock, P. J. A., Fields, C. J., Goto, N., Heuer, M. L. & Rice, P. M. The Sanger FASTQ file format for sequences with quality scores, and the Solexa/Illumina FASTQ variants. *Nucleic Acids Res.* **38**, 1767–1771 (2010).
19. Hedges, S. B. The origin and evolution of model organisms. *Nat. Rev. Genet.* **3**, 838–849 (2002).
20. Vöhringer, H. S. et al. Genomic reconstruction of the SARS-CoV-2 epidemic in England. *Nature* **600**, 506–511 (2021).
21. Loman, N. J., Quick, J. & Simpson, J. T. A complete bacterial genome assembled de novo using only nanopore sequencing data. *Nat. Methods* **12**, 733–735 (2015).
22. Aggarwal, D. et al. Genomic assessment of quarantine measures to prevent SARS-CoV-2 importation and transmission. *Nat. Commun.* **13**, 1012 (2022).
23. Ghafari, M. et al. Prevalence of persistent SARS-CoV-2 in a large community surveillance study. *Nature* **626**, 1094–1101 (2024).
24. Lythgoe, K. A. et al. Lineage replacement and evolution captured by 3 years of the United Kingdom Coronavirus (COVID-19) Infection Survey. *Proc. R. Soc. B: Biol. Sci.* **290**, 20231284 (2023).
25. Aggarwal, D. et al. Genomic epidemiology of SARS-CoV-2 in a UK university identifies dynamics of transmission. *Nat. Commun.* **13**, 751 (2022).
26. Elliott, P. et al. Dynamics of a national Omicron SARS-CoV-2 epidemic during January 2022 in England. *Nat Commun* **13**, 4500 (2022).
27. Garcia-Garcia, T. et al. Impairment of antiviral immune response and disruption of cellular functions by SARS-CoV-2 ORF7a and ORF7b. *iScience* **25**, 105444 (2022).
28. Liu, Y. et al. The N501Y spike substitution enhances SARS-CoV-2 infection and transmission. *Nature* **602**, 294–299 (2022).
29. Peter, A. S. et al. Characterization of SARS-CoV-2 Escape Mutants to a Pair of Neutralizing Antibodies Targeting the RBD and the NTD. *Int. J. Mol. Sci.* **23**, 8177 (2022).
30. De Coster, W. & Rademakers, R. NanoPack2: population-scale evaluation of long-read sequencing data. *Bioinformatics* **39**, btad311 (2023).
31. Bhattacharyya, A. On a Measure of Divergence between Two Multinomial Populations. *Sankhyā: Indian J. Stat.* **7**, 401–406 (1946).
32. Li, H. A statistical framework for SNP calling, mutation discovery, association mapping and population genetical parameter estimation from sequencing data. *Bioinformatics* **27**, 2987–2993 (2011).
33. Feng, Y. et al. COV2Var, a function annotation database of SARS-CoV-2 genetic variation. *Nucleic Acids Res* **52**, D701–D713 (2024).
34. Consortium, T. U. UniProt: the Universal Protein Knowledgebase in 2023. *Nucleic Acids Res* **51**, D523–D531 (2023).
35. Robinson, J. T. et al. Integrative genomics viewer. *Nat. Biotechnol.* **29**, 24–26 (2011).
36. O’Toole, Á. et al. Assignment of epidemiological lineages in an emerging pandemic using the pangolin tool. *Virus Evol.* **7**, veab064 (2021).
37. Aksamentov, I., Roemer, C., Hodcroft, E. & Neher, R. Nextclade: clade assignment, mutation calling and quality control for viral genomes. *J. Open Source Softw.* **6**, 3773 (2021).
38. Mencius, J. et al. Restoring flowcell type and basecaller configuration from FASTQ files of nanopore sequencing data. *Zenodo* <https://doi.org/10.5281/zenodo.15117614> (2025).
39. Stanojević, D., Lin, D., Florez de Sessions, P. & Šikić, M. Telomere-to-telomere phased genome assembly using error-corrected Simplex nanopore reads. *bioRxiv* 2024.05.18.594796. <https://doi.org/10.1101/2024.05.18.594796> (2024).
40. Poplin, R. et al. A universal SNP and small-indel variant caller using deep neural networks. *Nat. Biotechnol.* **36**, 983–987 (2018).
41. Shafin, K. et al. Haplotype-aware variant calling with PEPPER-Margin-DeepVariant enables high accuracy in nanopore long-reads. *Nat. Methods* **18**, 1322–1332 (2021).
42. Cleal, K. & Baird, D. M. Dysgu: efficient structural variant calling using short or long reads. *Nucleic Acids Res.* **50**, e53–e53 (2022).
43. Shiraishi, Y. et al. Precise characterization of somatic complex structural variations from tumor/control paired long-read sequencing data with nanomonsv. *Nucleic Acids Res* **51**, e74–e74 (2023).
44. Patterson, M. et al. WhatsHap: Weighted Haplotype Assembly for Future-Generation Sequencing Reads. *J. Comput. Biol.* **22**, 498–509 (2015).
45. Edge, P., Bafna, V. & Bansal, V. HapCUT2: robust and accurate haplotype assembly for diverse sequencing technologies. *Genome Res* **27**, 801–812 (2017).
46. Lin, J.-H., Chen, L.-C., Yu, S.-C. & Huang, Y.-T. LongPhase: an ultra-fast chromosome-scale phasing algorithm for small and large variants. *Bioinformatics* **38**, 1816–1822 (2022).
47. Kolmogorov, M. et al. metaFlye: scalable long-read metagenome assembly using repeat graphs. *Nat. Methods* **17**, 1103–1110 (2020).
48. Koren, S. et al. Canu: scalable and accurate long-read assembly via adaptive k-mer weighting and repeat separation. *Genome Res.* **27**, 722–736 (2017).
49. Ruan, J. & Li, H. Fast and accurate long-read assembly with wtdbg2. *Nat. Methods* **17**, 155–158 (2020).
50. Huang, Y.-T., Liu, P.-Y. & Shih, P.-W. Homopolish: a method for the removal of systematic errors in nanopore sequencing by homologous polishing. *Genome Biol.* **22**, 95 (2021).

Acknowledgements

This work was supported by the National Natural Science Foundation of China (No. 32370695 to Z.F.; No. 32470684 to H.F.) and the National Key R&D Program of China (No. 2022YFC2703400 to Y.Y., Z.F. and H.F.). Some of the computations in this work were run on the π 2.0 and Siyuan Mark-I clusters supported by the Center for High Performance Computing at Shanghai Jiao Tong University. Some of the computations in this research were performed using the CFF platform of Fudan University. We thank Chris Faulk (University of Minnesota) for kindly sharing the nanopore raw data of genome skimming datasets, Bernard Y. Kim (Stanford University) for kindly sharing the nanopore raw data of the *Drosophila melanogaster* dataset. We also thank Pengfei Wang, Xiaoyu Zhao, Xun Wang, and Cong Lai (all from Fudan University) for helpful discussions on COVID-19 genomics and epidemiology.

Author contributions

Z.F., H.F., K.S., and Y.Y. designed the project. Z.F., H.F., and J.M. devised the algorithms of LongBow. Z.F. and J.M. implemented the algorithm and developed the software. Y.Z. and J.M. collected publicly available nanopore raw datasets. Z.F., H.F., J.M., and T.A. designed and performed the evaluation of LongBow. J.M. and W.C. applied LongBow to restore the flowcell types and basecaller configurations of all public human nanopore sequencing data. J.M. and W.C. built the database of flowcell types and basecaller configurations. Z.F., H.F., and J.M. devised the algorithms of Artex. J.M. and W.C. implemented the algorithm and performed analysis of COG-UK data. Z.F. and H.F. wrote the manuscript with the help of J.M., W.C., Y.Z., K.S., and Y.Y..

Competing interests

Z.F., H.F., J.M., and W.C. are listed as authors on a patent application related to this work. The other authors declare no competing interests.

Additional information

Supplementary information The online version contains supplementary material available at <https://doi.org/10.1038/s41467-025-59378-x>.

Correspondence and requests for materials should be addressed to Yongguo Yu, Kun Sun, Huijuan Feng or Zhixing Feng.

Peer review information *Nature Communications* thanks the anonymous reviewer(s) for their contribution to the peer review of this work. A peer review file is available.

Reprints and permissions information is available at <http://www.nature.com/reprints>

Publisher's note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Open Access This article is licensed under a Creative Commons Attribution-NonCommercial-NoDerivatives 4.0 International License, which permits any non-commercial use, sharing, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons licence, and indicate if you modified the licensed material. You do not have permission under this licence to share adapted material derived from this article or parts of it. The images or other third party material in this article are included in the article's Creative Commons licence, unless indicated otherwise in a credit line to the material. If material is not included in the article's Creative Commons licence and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this licence, visit <http://creativecommons.org/licenses/by-nc-nd/4.0/>.

© The Author(s) 2025