Methods 234 (2025) 305–313

Contents lists available at ScienceDirect

Methods

journal homepage: www.elsevier.com/locate/ymeth

SV-JIM, detailed pairwise structural variant calling using long-reads and genome assemblies☆,☆

Clarence Todd [1], Lingling Jin [2,*], Ian McQuillan [3]

Department of Computer Science, University of Saskatchewan, Saskatoon, SK, Canada

ARTICLE INFO

Keywords:
Structural variant calling
Genetic variation
Comparative genomics

ABSTRACT

This paper proposes a detailed process for SV calling that permits a data-driven assessment of multiple SV callers that uses both genome assemblies and long-reads. The process is implemented as a software pipeline named Structural Variant – Jaccard Index Measure, or SVJIM, using the Snakemake [20] workflow management system. Like most state-of-the-art SV callers, SV-JIM detects the presence of variations between pairs of genomes, but it streamlines the numerous SV calling stages into a single process for user convenience and evaluates the multiple SV sets produced using the Jaccard index measure to identify those with the highest consistency among the included SV callers. SV-JIM then produces aggregated SV results based on how many callers supported the reported SVs. For validation, SV-JIM was assessed through three case studies on the Homo sapiens genome and two plant genomes – Brassica nigra and Arabidopsis thaliana. Executing SV-JIM identified a significant amount of inter-caller variance which varied by tens of thousands of results on the larger Brassica nigra and Homo sapiens genomes. Further, aggregating the SV sets helped simplify better retention of the less frequently occurring SV types by requiring a level of minimum support rather than from a specific SV caller combination. Finally, these case studies identified a potential for inflated precision reporting that can occur during evaluation. SV-JIM is available publicly under MIT license at https://github.com/USask-BINFO/SV-JIM.

1. Introduction

This paper studies the identification of large-scale genetic variations called *structural variants* (SVs) to mitigate several outstanding challenges that hinder their identification. SVs are variations that occur when the sugar-phosphate backbone of DNA is disrupted [2] and are classified into several types based on their rearrangement. SVs' size causes them to be more impactful on the content of a species' genome than smaller variations like single nucleotide polymorphisms (SNPs) due to the large-scale alterations they cause in a genome's coding regions and to its gene expression levels [1]. Consequently, SVs play a significant role in evolution and many genetic diseases, including cancer [18]. In plant species, SVs considerably impact yield and adaptability [7]. Therefore, opportunities exist for SV studies to help improve critical areas such as healthcare and global food security.

Traditionally, SV calling is performed using two methods involving sequence alignment. The first of these involves read alignment-based SV calling, which detects SVs based on discrepancies in how the reads of a query sample map back to a reference genome. This approach requires fewer computational steps to perform, but it often struggles with both repetitive regions and when detecting SVs that exceed current read lengths [19]. The second approach is performed by completing *de novo* assembly of the sequencing reads prior to comparing assemblies directly. Theoretically, this approach can detect all SV types; however, requiring *de novo* assembly can add considerably to the computational cost of this approach [17]. Errors made during assembly also affect downstream analyses, including SV calling. These errors occur more frequently in repetitive regions and can introduce sequence bias in high CG regions [16]. More recently, several other methods for SV calling have been introduced to improve the accuracy of SV calling, including

☆ This research is supported by the Natural Sciences and Engineering Research Council of Canada (NSERC), grant number 2019–06424 (LJ), 2022–05092 (IM), and a Canadian Graduate Scholarship – Doctoral (CMT).☆ This article is part of a special issue entitled: 'Disease-Related Omics Data Analysis' published in Methods.

* Corresponding author.
E-mail addresses: malcolm.todd@usask.ca (C. Todd), lingling.jin@cs.usask.ca (L. Jin), mcquillan@cs.usask.ca (I. McQuillan).
[1] 0000–0003–2522–2833.
[2] 0000–0002–4586–2347.
[3] 0000–0002–7998–4430.

https://doi.org/10.1016/j.ymeth.2024.12.015
Received 11 October 2024; Received in revised form 21 December 2024; Accepted 30 December 2024
Available online 16 January 2025

the use of deep learning for SV calling and SV calling from high throughput chromatin conformation capture (Hi-C) data. Currently, many deep learning-based callers detect only a subset of SV types, such as insertions and deletions only [17,4]. By contrast, Hi-C-based caller can detect the full range of SV types, but research into the identification of SVs using Hi-C data is still in the early stages with few tools currently available [15]. Therefore, this work focuses on SV callers employing the more traditional alignment-based approaches.

SV calling is now transitioning with the relatively recent introduction of long-read sequencing technologies and the increasing availability of reference-quality genome assemblies. Long-reads provide opportunities to improve alignment accuracy and SV calling sensitivity by leveraging their increased length to better span repetitive regions. However, low alignment accuracy still presents a significant challenge for any read alignment-based approach regardless of read size [10].

Different SV callers produce significant inconsistencies in their results when compared, even when using identical input alignments and settings. This has led to the creation of numerous SV callers that report varying accuracy rates on different SV types [12] and has prevented the creation of an SV caller that performs well in every situation. The inconsistency between SV callers remains challenging to evaluate and correct due to the lack of quality benchmark sets with which their accuracy could be assessed. SV callers are often benchmarked against the gold-standard Genome in a Bottle (GIAB) SV set for the human genome [27]. However, this set only features insertion and deletion SV locations and excludes more complex SV types like duplications, inversions, and translocations. Furthermore, the availability of gold-standard SV sets does not extend beyond the human genome. The lack of diversity in the available benchmarks has also caused many SV callers to prioritize the detection of SVs in the human genome during their development, creating a narrow research focus that has limited the exploration of SV calling effectiveness in other species.

This work proposes a software pipeline named SV-JIM to arm future SV studies with more tools that help leverage the benefits offered by long-reads and multiple reference genome assemblies, mitigate the variance in SV caller results, and work towards the accurately interpretating an SV's phenotypic impact. In addition, SV-JIM's results can help identify consistent and well-performing SV caller combinations for future studies. Finally, this work conducts SV experiments on species outside the human genome, which remain understudied.

2. Methodology

SV-JIM requires a pair of genome assemblies, with one sample serving as the reference genome and the other as the query. Additionally, a set of long-read data is required from the query sample for those SV callers that use them as input, and these reads should be those used to construct the query assembly when possible. SV-JIM features a four-stage process, including input preprocessing, sequence alignment, SV calling using SV calling using cuteSV [10], Sniffles [23], SVIM [7], PAV [5], and SVIM-asm [8], and the intersection and aggregation of the SV results. SV-JIM supports configuration through a single configuration file for improved convenience during setup, and its final output is a series of results from multiple combinations of the SV callers and evaluations performed using the Jaccard index measure. SV-JIM's detailed workflow is illustrated in Fig. 1, and a description of each stage's process is provided throughout this section.

2.1. Stages I and II: Input preprocessing and sequence alignment

During Stage I, SV-JIM performs input preprocessing by filtering the provided reference and query genomes to extract only the chromosomes or contigs of interest and can download any missing long-read files based on a user-provided list of accession numbers using SRA-Tools [24]. After preprocessing, SV-JIM performs sequence alignment with Minimap2 [13] between the query and reference genome assemblies and between the query genome's long-reads and the reference assembly. All SV callers SV-JIM uses accept Minimap2 alignments, guaranteeing that the results are generated from a shared baseline alignment. SV-JIM executes Minimap2 using its available sequencing technology and

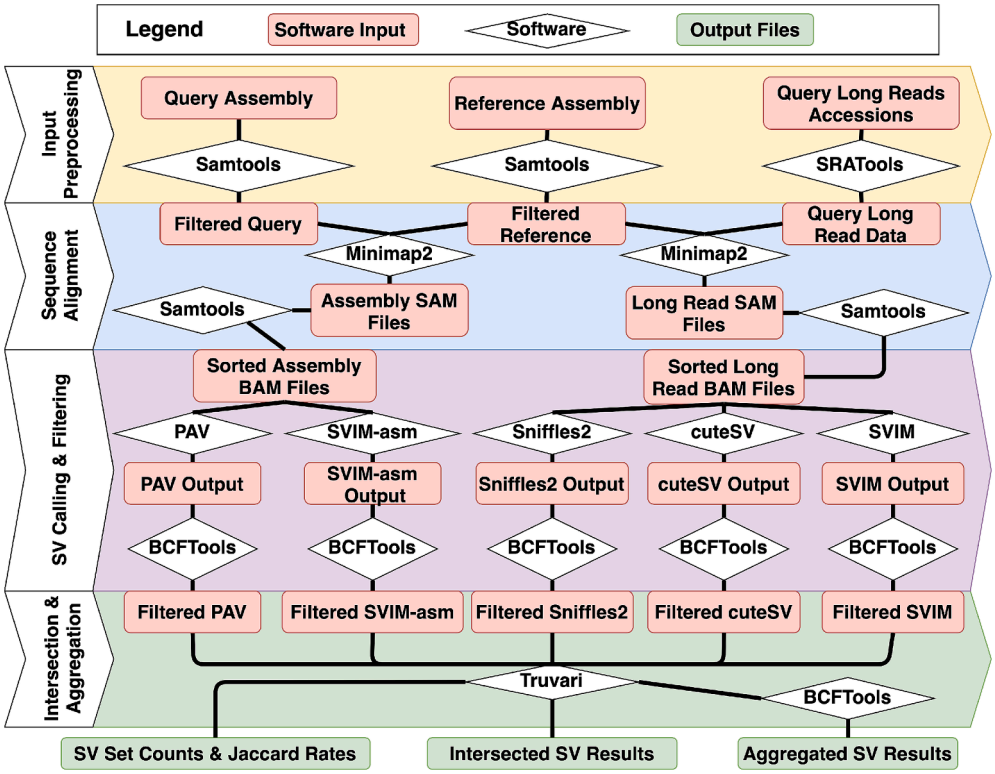


Fig. 1. An overview flowchart for SV-JIM's execution process.

assembly pre-sets. Minimap2′s sequencing read presets provide configurations for short single-end reads, Oxford Nanopore reads, and PacBio CLR, CCS, and HiFi reads [13]. Minimap2 also provides several assembly alignment preset configurations for aligning assemblies that diverge by up to five, ten, and twenty percent. Additionally, SV-JIM uses several Minimap2 settings from cuteSV's [10] and SVIM-asm's [8] publications by default; however, Minimap2 is configurable within SV-JIM. After alignment, SV-JIM sorts, indexes, and converts the output SAM alignment files to the BAM format using Samtools [3].

2.2. Stage III: SV calling and filtering for both long-read and genome assembly data

SV calling is performed using the long-read and genome assembly alignments produced during Stage II. SV-JIM incorporates three long-read callers named cuteSV [10], Sniffles2 [23], and SVIM [7], and two assembly-based SV callers named SVIM-asm [8] and PAV [5]. They were chosen for their recent development, citation history, and high reported performance. The ability to combine the benefits and strengths of the included SV callers provides *meta*-tools such as SV-JIM with valuable advantages, given the wider perspective created by the execution of a diverse set of algorithms and methods. Brief summaries of the methods and strengths of each SV caller are provided throughout this subsection; however, please consult each caller's original publication for more detailed information.

CuteSV is a long-read-based SV caller that extracts SV signature information from using CIGAR strings that were generated during alignment. CuteSV's execution is divided into three key stages, including multiple signature extraction for different SV types, grouping chimerically aligned reads through a cluster and refinement approach, and applying several tailored rules to perform SV calling and genotyping. CuteSV can support all sequencing technologies and provides the potential for better scaling to large datasets [10].

Sniffles2 is a long-read SV caller that uses within-alignment and split-read evidence to detect numerous SV types, including complex and nested SVs. Sniffles executes a five-stage process, including parameter estimation, scanning of read alignments to find SV locations, clustering and scoring SVs, optional SV genotyping, and optional clustering to detect complex SVs. The ability to identify complex SV makes Sniffles noteworthy because complex SVs like inverted tandem duplications and inversion flanked indels have yet to be studied to the same extent as more straightforward SV types [23].

SVIM is a long-read SV caller with a three-stage execution, including collecting SV signatures, signature clustering, and a final merging step. SVIM uses heuristics when searching for inter- and intra-alignment signatures by processing each aligned read to analyze its CIGAR strings and identify discordant read alignments that suggest the presence of SVs. SVIM also performs a more granular classification of large insertions into tandem duplications, interspersed duplications, or novel element insertions. Using this approach, SVIM reported higher precision and sensitivity than three other state-of-the-art SV callers, emphasizing its improved ability to distinguish between three types of insertions [8].

The Phased Assembly Variant caller, or PAV, is an assembly-based SV caller that supports phased and unphased genome assemblies. However, PAV only identifies insertion, deletion, and inversion SV types. PAV conducts several primary tasks during execution, including contig alignment and trimming, inter-alignment and alignment truncating variants, inversion detection, and call-set finishing. PAV also takes specific steps to improve inversion sensitivity by performing a *k*-mer density assessment using a *k*-mer size of 31 to build a density function for detecting SV locations and resolving the inner and outer breakpoints of flanking repeat sequences. PAV's approach proved effective in producing high-quality SV results since it generated a low false-positive rate of approximately 4 % when evaluated against GIAB's human SV benchmark set [5].

The Structural Variant Identification Method for Assemblies, or SVIM-asm, is an assembly-based SV caller with a similar workflow to SVIM. However, SVIM-asm features several modifications that leverage assembly alignment opportunities. Additionally, SVIM-asm can process both diploid and haploid assemblies. SVIM-asm can identify six distinct SV types, providing it advantages over other assembly-based callers that detect fewer SV types, such as PAV and DipCall [14]. SVIM-asm executes a four-stage process including signature extraction, signature pairing between haplotypes, genotyping, and reporting the genotyped SVs classified into one of the supported types. That said, SVIM-asm employs a simplified workflow when provided with haploid genomes by skipping the pairing and genotyping stages. SVIM-asm demonstrated improved F1 scores and genotyping accuracy over DipCall using GIAB's human benchmark SV set due to a lower false-positive rate and increased recall with large insertions and deletions that DipCall ignored [8].

During execution, the read-based SV callers are configured using parameters outlined in cuteSV's Supplementary Materials [10] because they were used to compare the same three long-read SV callers; however, SV-JIM itself also permits configuring their parameters. These configurations filter results based on their maximum size, alignment quality (MAPQ), and amount of read evidence observed. The maximum length threshold prevents the inclusion of SVs spanning millions of bps or the majority of chromosomes assumed to be erroneous. Likewise, the quality scores are used to restrict the SV result's probability of error, and including the minimum number of supporting reads controls the amount of evidence necessary for the read-based callers to report SVs. Details on SV-JIM's default settings are provided in the Supplementary Information.

These settings improve the confidence of the predicted SV results and aid SV-JIM in producing a higher calibre of SV results; however, most of the included SV callers do not provide configuration options for all the required filtering. To improve the consistency and the compatibility of the SV callers' results, SV-JIM performs additional filtering using BCFtools [3] to enforce as many filter settings as possible, but differences in each caller's VCF reporting prevented the use of a consistent filtering expression.

After filtering, the SV results are partitioned into separate sets according to their SV type. This allows a more granular comparison of the SV callers' performance when identifying and quantifying similarities and differences between their results. Additionally, this partitioning provides greater flexibility in the outputs generated should users be interested in SV results of specific types.

2.3. Stage IV: SV result intersection and aggregation

In the final stage, the segmented VCF files are combined using Truvari [6] to calculate the intersection between all combinations of the SV callers, including all sets of two, three, four, and five callers. These intersection sets identify the SVs common to multiple callers or unique to one. SV-JIM identifies SVs that share greater than 70 % size similarity and are reported within 2000bps of each other in the reference genome. These configurations were selected because they were recommended by Zook et al. [27] when comparing SV results to their published gold standard human SV set.

Finally, SV-JIM combines the intersecting SV sets using BCFtools to produce aggregated SV sets for each caller based on the minimum number of supporting callers. This aggregation step yielded valuable insights regarding how much the other callers supported each SV caller's results. Additionally, this approach enables evaluating a different SV set combination strategy similar in nature to the process used in constructing the gold-standard GIAB benchmark set itself, where SVs required support from two or more technologies, five or more SV callers, or the Bionano/Nabsys data used [27].

2.4. Performance evaluation

SV-JIM calculates the Jaccard similarity coefficient [9], or Jaccard

index, as a performance measure for inter-caller consistency. The Jaccard index is a measure of similarity between two sets of SVs detected by two callers, ranging from 0 % to 100 %, calculated as follows:

$$J(A,B) = \frac{|A \cap B|}{|A \cup B|} \quad (1)$$

The performance of each SV caller can then be compared using the Jaccard index by treating the shared outputs as valid SV results and unique calls as errors. This evaluation is performed on all twenty-six of the combinations of at least two SV callers generated during SV-JIM's execution to capture changes in consistency as the number of SV callers involved increases and to allow an average of the Jaccard indices to serve as an overall performance score for each caller. Likewise, assessing all twenty-six combinations involving the five included SV callers helps to ensure a thorough evaluation of their consistency. This approach enables SVJIM to mitigate a single SV caller's inability to capture all SVs accurately by generating a more consistent output of SVs in the sample.

The GIAB benchmark partitions the benchmark SV set into tiers based on the confidence of the predicted SV's location. The Tier 1 regions contain nearly 100 % accurate insertions and deletions, and Tier 2 regions show strong evidence of an SV, but whose size cannot be determined confidently [27]. Any predicted SVs that did not at least match these tiers' criteria were deemed to be of insufficient confidence. Using Truvari, the SV caller's precision, recall, and F1-scores were calculated to study their accuracy, and they are calculated as follows:

$$Precision = \frac{TruePositives}{TruePositives + FalsePositives} \quad (2)$$

$$Recall = \frac{TruePositives}{TruePositives + FalseNegatives} \quad (3)$$

$$F1 = \frac{2{*}Precision{*}Recall}{Precision + Recall} \quad (4)$$

The results from SV-JIM were compared with the Tier 1 & 2 regions defined by GIAB, and predicted SVs reported within 2000 bps of the GIAB SV with greater than 70 % size similarity are considered correct predictions.

3. Results and discussion

3.1. Individual SV caller results

SV-JIM was validated through three experiments using genome samples from *Brassica nigra*, *Arabidopsis thaliana*, and *Homo sapiens*. These experiments were executed on a Dell R740 server featuring 1.5 TB of RAM, 44 TB of storage, and Dual Intel Xeon Gold 5220 CPUs with 36 cores and 72 hyperthreads. Most of SV-JIM's resource demands are generated by a subset of its included tools with PAV, Minimap2, SVIM, and Samtools being responsible for more than 90 % of the total execution time during these experiments. Therefore, users are encouraged to consult these tools' online resources to locate up-to-date information concerning their computational demands. Details on the experiments' samples and execution times are provided in Table 1, and the required execution times for the four software tools above are provided in the Supplementary Materials. All three case studies were conducted using the same configurations across all tools to improve the consistency of their results. These settings represent SV-JIM's default settings, with the commands used for each tool provided in Table 2.

There were significant differences in the outputs from each SV caller when provided the same input data and configurations, which varied by thousands of SVs using the *A. thaliana* data and tens of thousands with the *B. nigra* and *H. sapiens* data. A portion of this variance could be explained by SV-JIM's inability to apply consistent filtering to all SV callers due to differences in their VCF reporting, including some callers' overuse of placeholder values in key data fields. Flexibility is a key strength of the VCF format; however, setting a more stringent expectation for VCF reporting in future callers could be a valuable step towards reducing their results' variance. The results also revealed that most of the SVs detected were insertions and deletions ranging from 100 bps to 1 Kbp in size. By contrast, significantly fewer duplications and inversions were detected, and their size typically ranged from 1 Kbps to 100 Kbps. A summary of the individual SV caller totals is provided in Fig. 2.

3.2. Intersection SV set results

The individual SV caller results were compared using Truvari to generate 26 intersected combinations of the SV callers, representing all sets of two, three, four, and five callers. These intersections then allowed for calculating each combination's Jaccard index ratio to quantify the consistency rate between the included callers. There were significant differences in the amount of consensus between the callers when intersected, as suggested by the differences in their relative sizes. Based on the Jaccard index ratios received, deletions and insertions were identified with significantly higher consistency between callers than the duplication, inversion, and translocation breakend types. There were even instances in the *A. thaliana* results where some of the SV callers did not identify any results for these types, strongly impacting the intersection sets produced.

Sniffles and cuteSV were most consistent in their results from all pairs of callers for both plant species and cuteSV and SVIM were the most consistent pair using the *H. sapiens* data. By contrast, the most divergent caller varied for each species. Intersecting pairs of callers also revealed many instances where the smaller SV caller's results demonstrated a high overlap rate with the larger caller's results. This indicates that some results may reflect differences in the sensitivity of their underlying algorithms and that taking the consensus from even a small number of callers can quickly narrow SV results to a well-supported set. However, this overlapping relationship is typically one-way between callers, which introduces the consideration that taking consensus from certain combinations could significantly reduce the overall sensitivity of the SV results. Therefore, exploring alternative methods to combine SV results, like aggregation, could better balance the reduction in

Table 1
Details for the samples and wall clock times from SV-JIM's experiments.

Species	Reference Sample	Query Sample	Data Source	Threads	Duration
A. thaliana	TAIR10 [25]	Ler [11]	1001Genomes.org	30	2H:48 M:8S
B. nigra [21]	NI100	C2	crucieferseq.ca	30	1D:21H:29 M:7S
H. sapiens	hs37d5	HG002 [26]	ncbi.nlm.nih.gov	50	1D:13H:57 M:56S

Table 2
SV caller commands executed by SV-JIM. Parameter values in these commands can be modified using the provided configuration file. *PAV executes as a separate Snakemake pipeline, which was executed using its default settings.

SV caller	Executed commands
cuteSV	cuteSV –min –size 50 –max size 300,000 –s 10 –max –cluster bias –INS 100 –diff ratio merging INS 0.3 –max cluster bias DEL 100 –diff ratio merging DEL 0.3 {inputFile}
PAV*	bash./Scripts/run-PAV-SnakeMake-pipe.sh {threads}{svOutputDir} {input}{queryID}
Sniffles2	sniffles –threads 30 –mapq 20 –minsupport 10 –minsvlen 50 –reference {refGenome} –i {inputFile} –v {output}
SVIM	svim alignment –min sv size 50 –max sv size 300,000 –min mapq 20 {svOutputDir} {inputFile}{refGenome}
SVIM-asm	svim-asm haploid –min mapq 20 –min –sv size 50 –max sv _size 300,000 {svOutputDir} {inputFile}{refGenome}

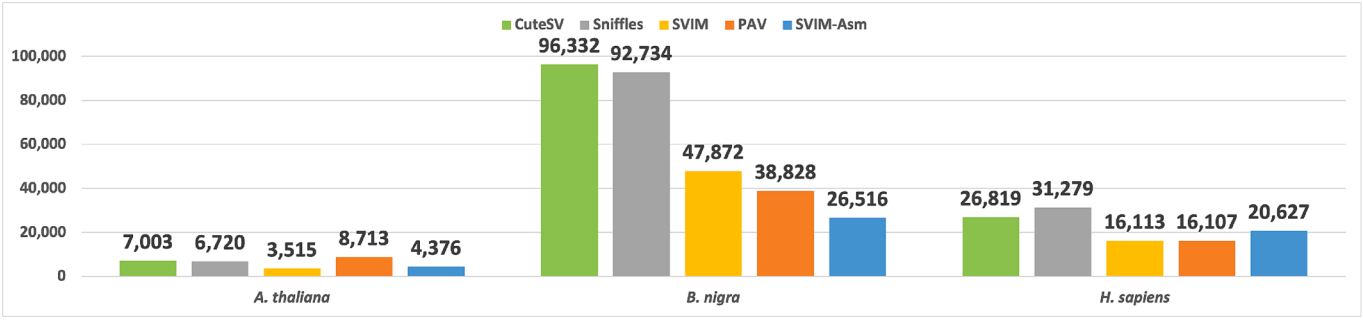


Fig. 2. SV caller totals from all the species used during validation.

sensitivity while still allowing for the identification of SV results supported by multiple callers. There were also differences in each caller's relative performance between experiments, indicating the different species' genomes affected each SV caller's performance differently. This species-specific influence suggests that each SV caller may require fine-tuning based on the input sequences to improve its results. The amount of difference between each pair of SV caller's results from all experiments is illustrated in Fig. 3.

There were significant differences in the combinations of three callers that shared the most similar results in each experiment. For example, cuteSV, Sniffles2, and SVIM were most consistent for *B. nigra* and cuteSV, PAV, and Sniffles2 were most similar for *A. thaliana*. By contrast, PAV, Sniffles2, and SVIM-asm were the most similar *H. sapiens* combination. Likewise, there were differences in the most similar combinations of four SV callers with cuteSV, PAV, Sniffles2, and SVIM for *B. nigra*; however, both *A. thaliana* and *H. sapiens* reported the most similarity between cuteSV, PAV, Sniffles2, and SVIM-asm. The combinations of more SV callers demonstrated higher consistency in *H. sapiens* than they did in the plant genomes. When applied to the human genome, this higher consistency between callers may have been caused by its involvement in many callers' validation during development, thereby making them better tuned to its contents. However, this demonstrates the value of conducting further studies in plant genomes to experiment with these callers in contexts with more repetitive elements and different ploidy levels. Finally, the combination of all five callers retained a tiny subset of the callers' original results. The distributions of the SV counts for the callers' intersections are illustrated in Fig. 4.

Overall, the Jaccard index measure proved very useful when no gold-standard benchmark set was available to calculate reliable precision and recall, and it added further perspective to the assessment. However, the size of the input SV sets influences the measure, and it may be biased against larger or more sensitive results due to its reliance on majority belief among callers when determining correct results. Further, the Jaccard index cannot assess accuracy components like false negatives, preventing it from replacing metrics like precision and recall.

3.3. Aggregation SV set results

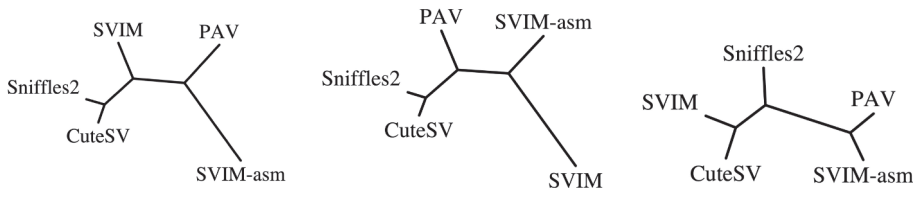
After the intersection of all SV caller combinations, their intersections were aggregated into SV sets for each caller based on how many others supported the reported SVs. This provided more flexibility in how results were retained and additional perspective into the inconsistency between SV caller results. For each caller, aggregated sets for SVs supported by any two or more, three or more, and four or more supporting callers were generated as SV-JIM's final output. These aggregated sets demonstrated improved retention of the duplication and inversion results by moderating the consensus required between callers versus those retained within many of the specific intersections. The results showed that SVIM provided the best retention of its results across all three species and that the SV caller with the lowest retention varied between species. That said, SVIM frequently produced the smallest SV result in the experiments, which may have caused its results to influence which results were retained from the other callers. The retention rates appear high overall; however, the high proportion of insertions and deletions overshadows the substantially lower retention of the other more complex SV types in these sets. Details regarding the aggregated sets' retention rates are illustrated in Fig. 5.

3.4. SV benchmarking results

After execution, the deletion and insertion SVs from the individual, intersected, and aggregated results produced by SV-JIM were benchmarked against the Tier I and II locations identified in the GIAB benchmark set [27] using Truvari. Three benchmarking experiments were performed on the individual SV caller results with the Tier I and Tier I & II SVs, including:

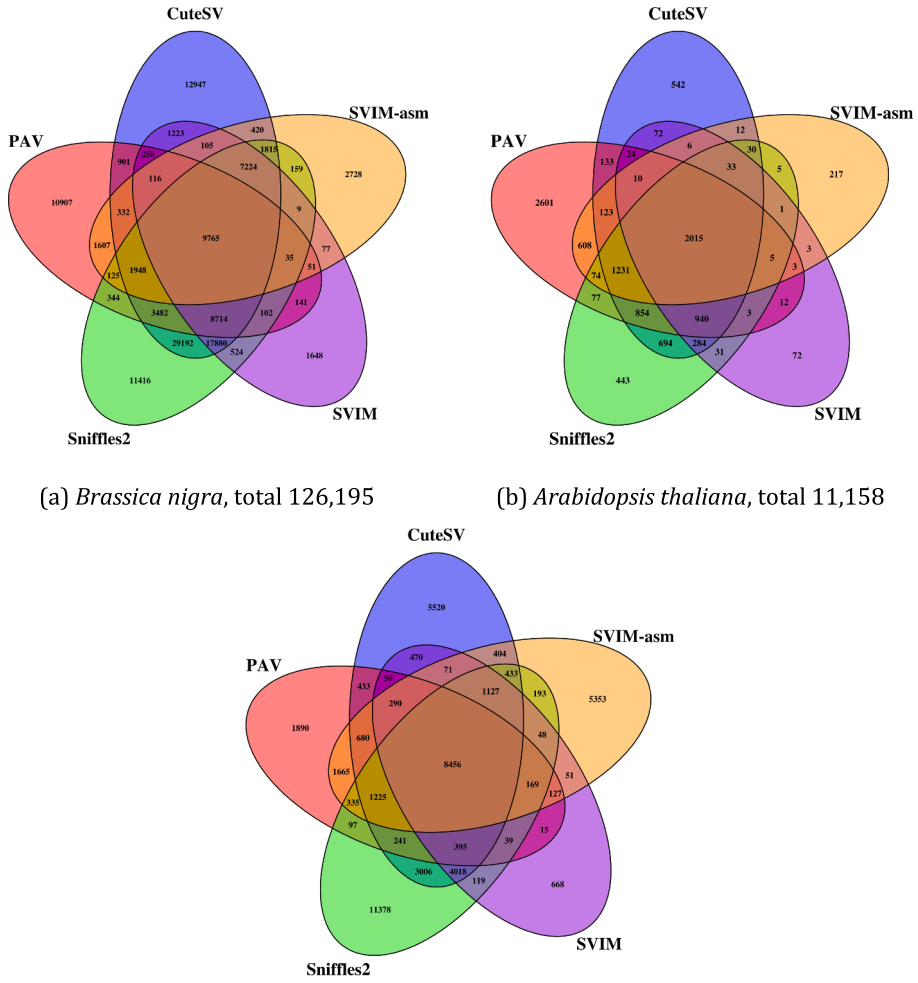
1. Prefiltering the benchmark using bedtools [22]
2. Providing Truvari the Tier I & II locations to perform prefiltering
3. Providing Truvari just the Tier I locations to perform prefiltering

Benchmarking the individual SV caller results provided a baseline assessment of each caller's performance. Using the benchmark produced by bedtools, most callers provided strong recall but lower precision. Prioritization of recall over precision can be explained by the clinical or disease-related applications SV callers can be used for and the high sensitivity they demand. The highest recall rate was provided by cuteSV, and SVIM provided the highest precision rate and F1 score. By contrast,



(a) *Brassica nigra* (b) *Arabidopsis thaliana* (c) *Homo sapiens*

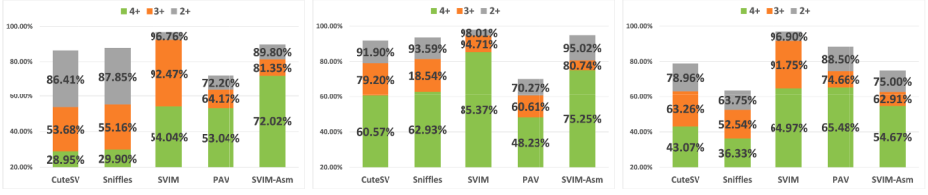
Fig. 3. Tree representation of the relationship between SV callers based on the differences in their results. Each branch length represents the pair's *Jaccard distance*, or the sets' rate of difference, calculated as $1-J(A,B)$.



(a) *Brassica nigra*, total 126,195 (b) *Arabidopsis thaliana*, total 11,158

(c) *Homo sapiens*, total 48,966

Fig. 4. Venn diagrams for the intersecting distributions from all experiments.

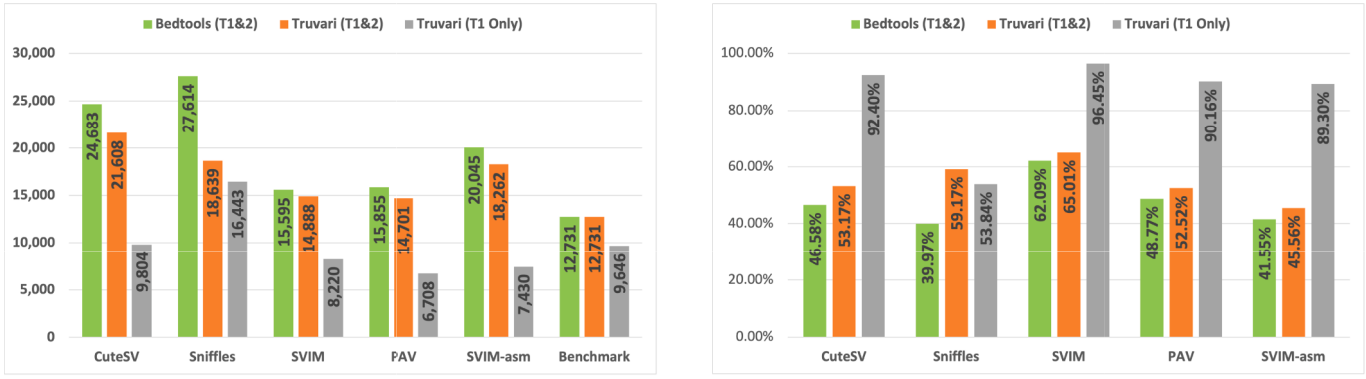


(a) *Brassica nigra* (b) *Arabidopsis thaliana* (c) *Homo sapiens*

Fig. 5. SV caller retention rates from all species' aggregated sets. Rates are calculated as the aggregated set's size divided by the original caller result's size.

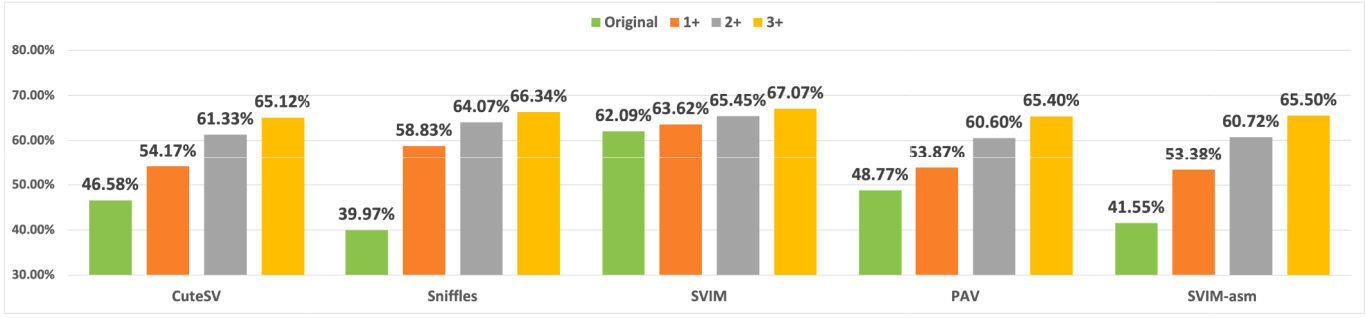
Truvari's benchmarking, when provided the Tier I BED regions, reported significantly higher precision rates of around 90 %. The primary reason for this is its prefiltering on the input query SV set, which removes many false positive results from the evaluation. The prefiltering effect is observable in the reduction of the SV caller totals and the significant increase in the precision reported when Truvari is provided with a BED file. This represents a concern for evaluating SV callers since they are intended to locate SVs without *a priori* knowledge of their locations, meaning this prefiltering skews the evaluation's reliability. The impacts of Truvari's prefiltering on the SV caller and benchmark sets' total sizes and their resulting precision are detailed in Fig. 6.

Benchmarking the intersections for all combinations of SV callers provided insights into the effect of SV caller consensus on precision and recall. These experiments also identified several high-performing combinations using their reported F1 scores. The F1 score was used in this analysis due to its property as the harmonic mean of precision and recall, such that any improvement in the F1 score was seen as productive regardless of the specific changes in the two underlying metrics. For example, cuteSV and Sniffles2 produced the best F1 scores among all pairs of callers at 70.15 %. Likewise, the best F1 score from the combinations of three callers was produced by cuteSV, Sniffles2, and SVIM at 70.27 %. Finally, cuteSV, Sniffles2, SVIM, and SVIM-asm produced the best F1 score among the combinations of four callers at 59.46 %, representing a notable drop relative to those achieved by the combinations of three. That said, the precision achieved by combinations of four callers continued to rise despite the drop in F1 scores, so an argument could

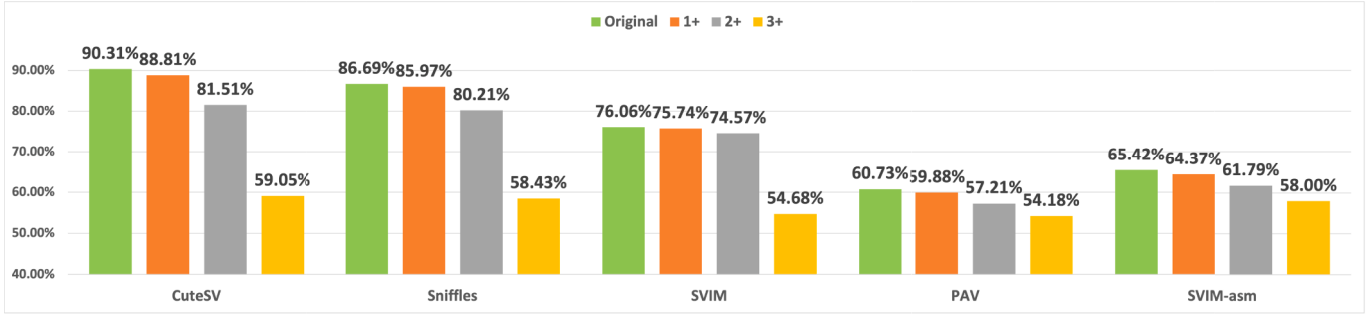


(a) Filtered result totals (b) Reported precision rates

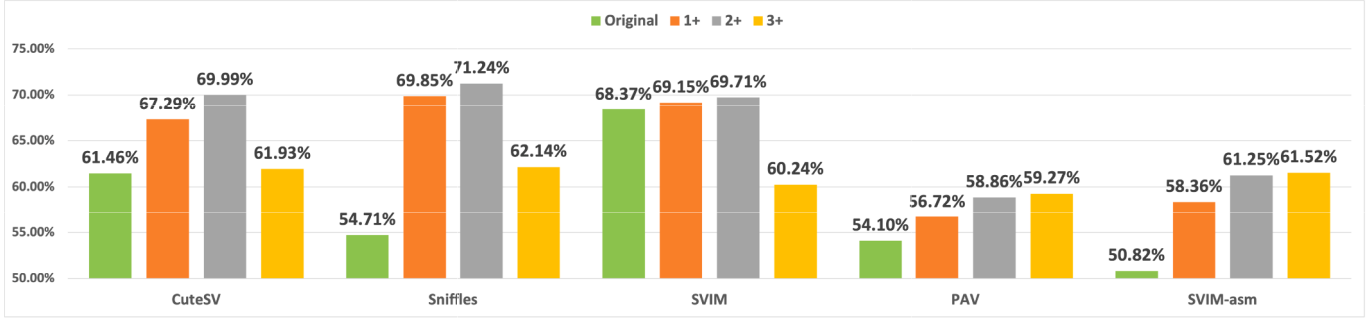
Fig. 6. Effect of prefiltering performed by either Bedtools or Truvari on SV caller and benchmark total sizes and reported precision. The bedtools caller totals are the originals untouched by prefiltering. Truvari pre-filtering tested using BED files containing regions from both Tier 1 and Tier 2 (T1&2) or Tier 1 only (T1).



(a) Precision



(b) Recall



(c) F1 Score

Fig. 7. Precision, recall, and F1 scores observed from the aggregated sets' benchmarking. N+ value denotes the required support from another N or more callers.

be made that these combinations continue to strengthen the quality of the results if high precision were a study's primary objective.

Finally, benchmarking each caller's aggregated set revealed the relationship between precision and recall, a tradeoff in favour of precision as the number of required supporting callers increases. The nature of this relationship can be observed in the increasing trend in caller precision versus the corresponding drop in recall as callers are added. However, this relationship experienced significantly diminishing returns with each caller added and achieved the highest F1 scores when aggregating support from three callers. This was also consistent with the observation made when comparing the best F1 scores for each combination size among the intersection sets. The genome assembly-based SV callers continue to see marginal improvements when adding a fourth supporting caller. Therefore, further experimentation is required to determine if this trend applies beyond these experiments. From the results, the best precision was achieved by SVIM when aggregated with any three other supporting callers and the best F1 score of 71.24 % was provided by Sniffles2 when aggregated with any two other callers. This trade-off between precision and recall partially explains the higher performance the read-based SV callers reported since they often had more recall to trade for precision as callers were added. Further, this demonstrates the value of incorporating SV callers that take different types of input data to improve the SV results. Details regarding the aggregated sets' precision and recall performance are provided in Fig. 7.

Evaluating the SV results provided valuable perspectives into the current challenges for benchmarking SV callers, including limitations in the number of available benchmarks and the concerns created by some of the prefiltering used by popular SV benchmarking software. SV-JIM's *B. nigra* results also identified low overlap rates between each SV caller; however, no further assessment could be conducted given the lack of a gold-standard benchmark. Additionally, the high variance in the callers' results suggests that benchmarking them in multiple contexts is necessary to determine if they can perform well in all situations. Therefore, creating additional benchmarks outside the human genome represents a critical research direction to improve SV calling. Truvari's prefiltering also raises several questions about the reliability of existing callers' performance evaluations if their benchmarking excluded false positives by leveraging SV location information known *a priori*. Consequently, further research into prefiltering alternatives that produce the same filtered benchmark as Truvari without influencing the SV caller inputs could help to improve current benchmarking efforts.

4. Conclusions

This paper proposes a Snakemake [20] pipeline named SV-JIM that compares and combines multiple existing SV callers' results to conduct a comparative study by analysing inter-caller variance and consensus's effect on the caller's results. SV-JIM was validated through experiments using *Brassica nigra*, *Arabidopsis thaliana*, and *Homo sapiens* genome data. Executing SV-JIM provided several insights, including the differences in the SV caller's results and built-in filtering, several strengths and limitations of using the Jaccard index measure for inter-caller variance, the benefits of aggregating SV sets, and some current challenges in benchmarking SV accuracy. Additionally, combining SV caller results appears to achieve the optimum balance in precision and recall when requiring the support of three SV callers, based on their F1 scores.

These experiments revealed several future development directions for SVJIM, including adding additional workflows, such as the optional execution of Hi-C-based tools for cases where users can provide this type of data. Additionally, the inclusion of deep learning-based tools could be revisited in the future as they provide better variety in the types of SVs they detect. Beyond SV-JIM, these experiments also revealed several directions for future research, including developing SV benchmarks for more species, reviewing current benchmarking practices to avoid influencing caller evaluation, and further exploring SV caller's results on multiple species to identify which genetic features produce differences in their accuracy.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Appendix A. Supplementary data

Supplementary data to this article can be found online at https://doi.org/10.1016/j.ymeth.2024.12.015.

Data availability

All data used within this work in available publicly at the provided links in the manuscript and Supplementary Files. The proposed software is also available through GitHub at the link provided.

References

[1] M. Alonge, X. Wang, M. Benoit, S. Soyk, L. Pereira, L. Zhang, H. Suresh, S. Ramakrishnan, F. Maumus, D. Ciren, et al., Major impacts of widespread structural variation on gene expression and crop improvement in tomato, Cell 182 (1) (2020) 145–161.
[2] C.M. Carvalho, J.R. Lupski, Mechanisms underlying structural variant formation in genomic disorders, Nat. Rev. Genet. 17 (4) (2016) 224–238.
[3] P. Danecek, J.K. Bonfield, J. Liddle, J. Marshall, V. Ohan, M.O. Pollard, A. Whitwham, T. Keane, S.A. McCarthy, R.M. Davies, H. Li, Twelve years of SAMtools and BCFtools, Giga-Science 10 (2) (2021) giab008, https://doi.org/10.1093/gigascience/giab008.
[4] H. Ding, J. Luo, MAMnet: detecting and genotyping deletions and insertions based on long reads and a deep learning approach, Brief. Bioinform. 23 (5) (2022) bbac195.
[5] P. Ebert, P.A. Audano, Q. Zhu, B. Rodriguez-Martin, D. Porubsky, M.J. Bonder, A. Sulovari, J. Ebler, W. Zhou, R. Serra Mari, et al., Haplotype-resolved diverse human genomes and integrated analysis of structural variation, Science 372 (6537) (2021) eabf7117.
[6] A.C. English, V.K. Menon, R.A. Gibbs, G.A. Metcalf, F.J. Sedlazeck, Truvari: refined structural variant comparison preserves allelic diversity, Genome Biol. 23 (1) (2022) 1–20.
[7] D. Heller, M. Vingron, SVIM: structural variant identification using mapped long reads, Bioinformatics 35 (17) (2019) 2907–2915.
[8] D. Heller, M. Vingron, SVIM-asm: structural variant detection from haploid and diploid genome assemblies, Bioinformatics 36 (22–23) (2020) 5519–5521.
[9] P. Jaccard, The distribution of the flora in the alpine zone. 1, New Phytol. 11 (2) (1912) 37–50.
[10] T. Jiang, Y. Liu, Y. Jiang, J. Li, Y. Gao, Z. Cui, Y. Liu, B. Liu, Y. Wang, Long-read-based human genomic structural variation detection with cuteSV, Genome Biol. 21 (1) (2020) 1–24.
[11] W.B. Jiao, K. Schneeberger, Chromosome-level assemblies of multiple *Arabidopsis* genomes reveal hotspots of rearrangements with altered evolutionary dynamics, Nat. Commun. 11 (1) (2020) 1–10.
[12] S. Kosugi, Y. Momozawa, X. Liu, C. Terao, M. Kubo, Y. Kamatani, Comprehensive evaluation of structural variation detection algorithms for whole genome sequencing, Genome Biol. 20 (1) (2019) 117.
[13] H. Li, Minimap2: pairwise alignment for nucleotide sequences, Bioinformatics 34 (18) (2018) 3094–3100.
[14] H. Li, J.M. Bloom, Y. Farjoun, M. Fleharty, L. Gauthier, B. Neale, D. MacArthur, A synthetic-diploid benchmark for accurate variant-calling evaluation, Nat. Methods 15 (8) (2018) 595–597.
[15] J. Li, L. Gao, Y. Ye, HiSV: a control-free method for structural variation detection from Hi-C data, PLoS Comput. Biol. 19 (1) (2023) e1010760.
[16] X. Liao, M. Li, Y. Zou, F.X. Wu, J. Wang, Current challenges and solutions of de novo assembly, Quant. Biol. 7 (2) (2019) 90–109.
[17] J. Luo, H. Ding, J. Shen, H. Zhai, Z. Wu, C. Yan, H. Luo, BreakNet: detecting deletions using long reads and a deep learning approach, BMC Bioinf. 22 (2021) 1–13.
[18] G. Macintyre, B. Ylstra, J.D. Brenton, Sequencing structural variants in cancer for precision therapeutics, Trends Genet. 32 (9) (2016) 530–542.
[19] M. Mahmoud, N. Gobet, D.I. Cruz-Dávalos, N. Mounier, C. Dessimoz, F. J. Sedlazeck, Structural variant calling: the long and the short of it, Genome Biol. 20 (1) (2019) 246.
[20] F. Mölder, K.P. Jablonski, B. Letcher, M.B. Hall, C.H. Tomkins-Tinch, V. Sochat, J. Forster, S. Lee, S.O. Twardziok, A. Kanitz, et al., Sustainable data analysis with Snakemake, F1000 Research 10 (2021).
[21] S. Perumal, C.S. Koh, L. Jin, M. Buchwaldt, E.E. Higgins, C. Zheng, D. Sankoff, S. J. Robinson, S. Kagale, Z.K. Navabi, et al., A high-contiguity *Brassica nigra* genome localizes active centromeres and defines the ancestral *Brassica* genome, Nat. Plants 6 (8) (2020) 929–941.

[22] A.R. Quinlan, I.M. Hall, BEDTools: a flexible suite of utilities for comparing genomic features, Bioinformatics 26 (6) (2010) 841–842.
[23] F.J. Sedlazeck, P. Rescheneder, M. Smolka, H. Fang, M. Nattestad, A. Von Haeseler, M.C. Schatz, Accurate detection of complex structural variations using single-molecule sequencing, Nat. Methods 15 (6) (2018) 461–468.
[24] Team, S.T.D.: NCBI/SRA-Tools: SRATools, https://github.com/ncbi/sra-tools.
[25] D. Weigel, R. Mott, The 1001 genomes project for *Arabidopsis thaliana*, Genome Biol. 10 (5) (2009) 1–5.
[26] J.M. Zook, D. Catoe, J. McDaniel, L. Vang, N. Spies, A. Sidow, Z. Weng, Y. Liu, C. E. Mason, N. Alexander, et al., Extensive sequencing of seven human genomes to characterize benchmark reference materials, Sci. Data 3 (1) (2016) 1–26.
[27] J.M. Zook, N.F. Hansen, N.D. Olson, L. Chapman, J.C. Mullikin, C. Xiao, S. Sherry, S. Koren, A.M. Phillippy, P.C. Boutros, et al., A robust benchmark for detection of germline large deletions and insertions, Nat. Biotechnol. 38 (11) (2020) 1347–1355.