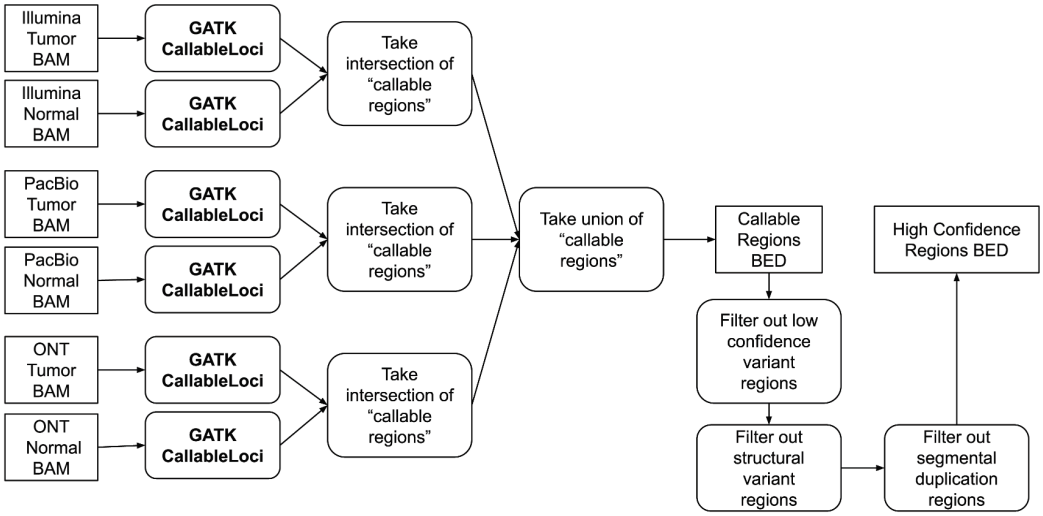
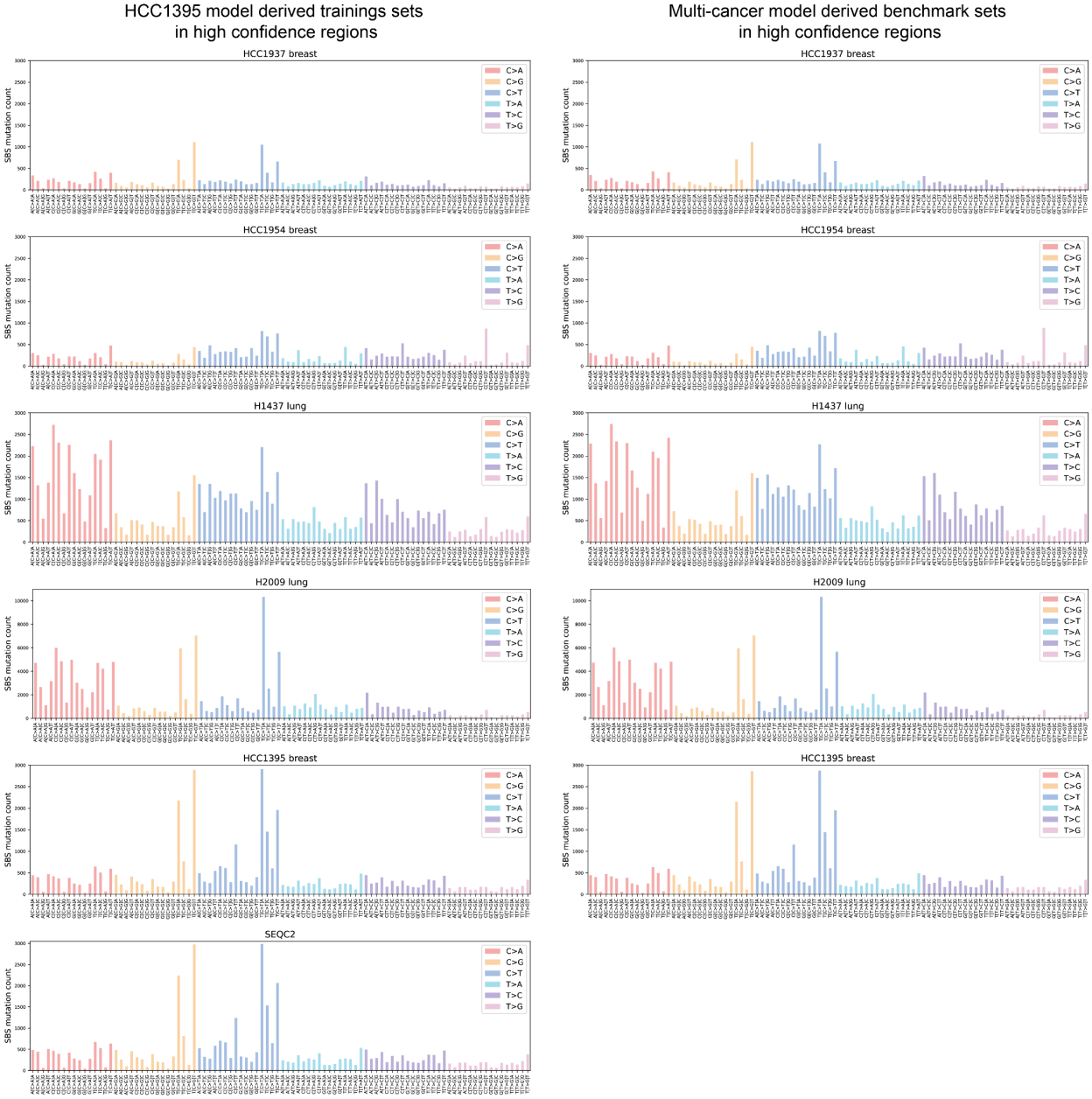
Supplementary

Supplementary tables

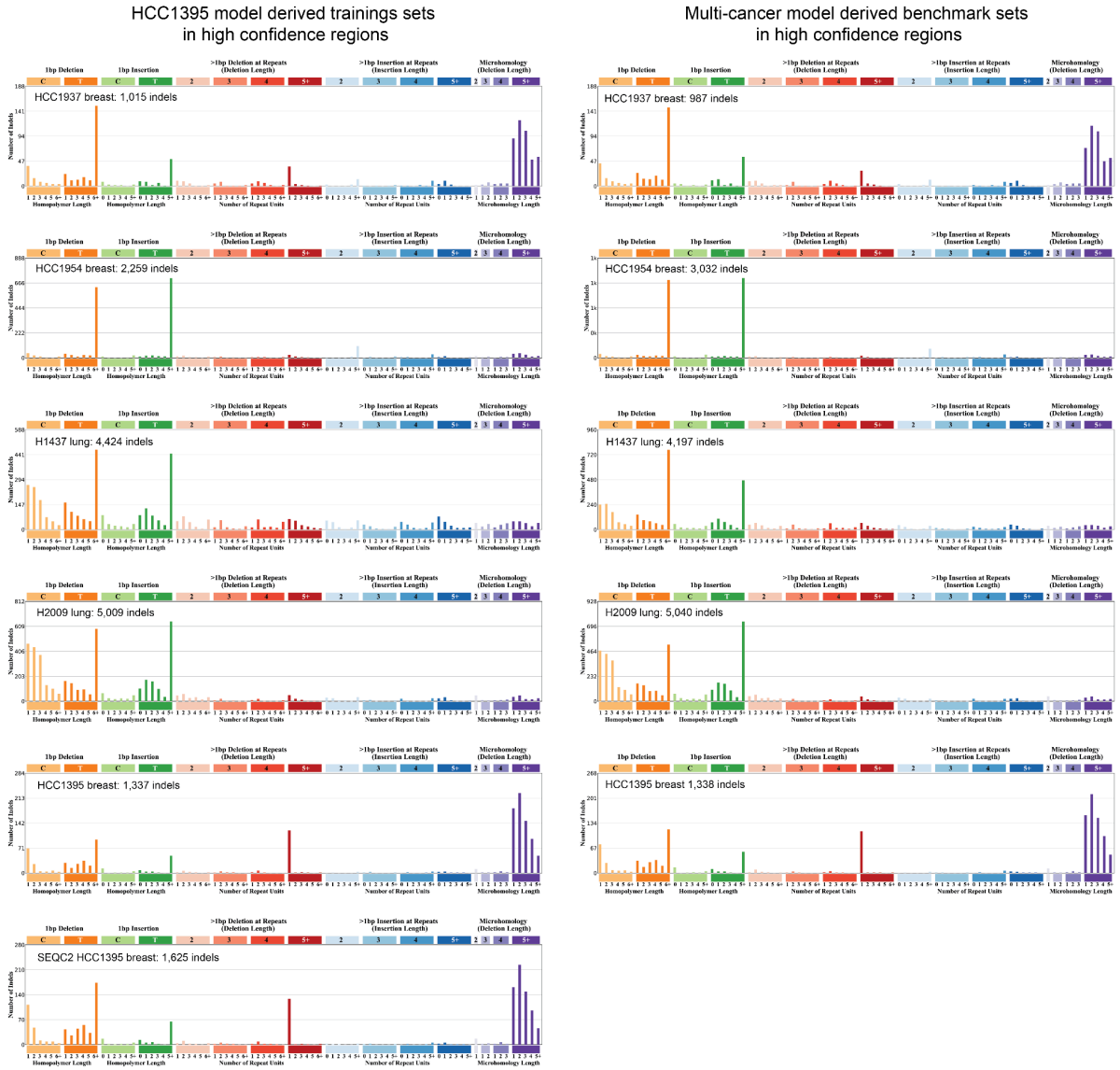
Supplementary figures



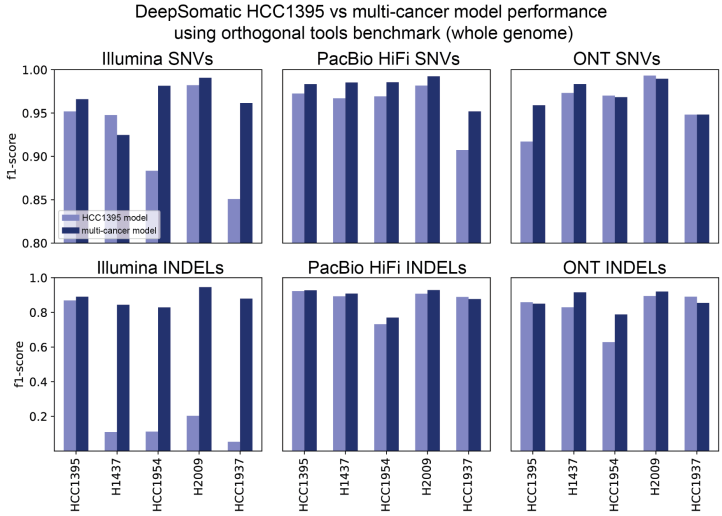
Supplementary Figure 1: Steps for generating high confidence regions BED files.



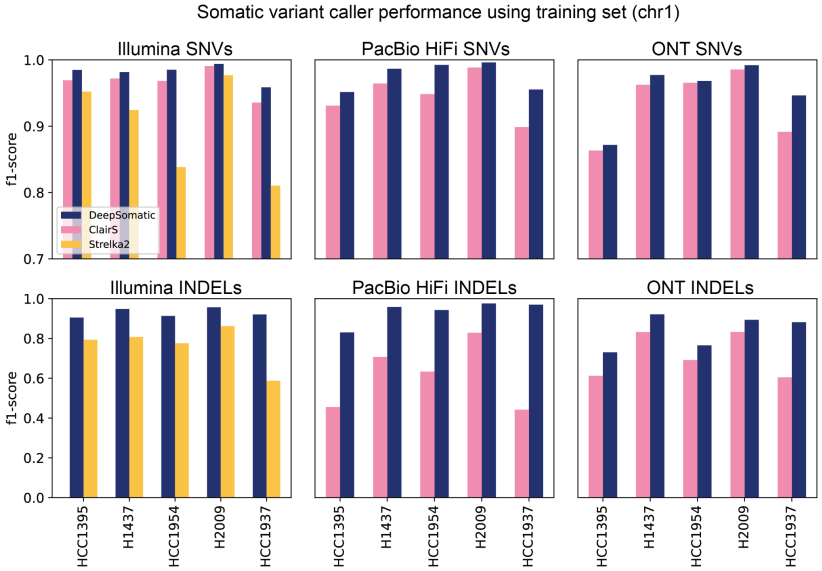
Supplementary Figure 2: SBS-96 counts for SEQC2 benchmark and each benchmark set derived from DeepSomatic HCC1395 models (left) and multi-cancer models (right) in high-confidence regions.



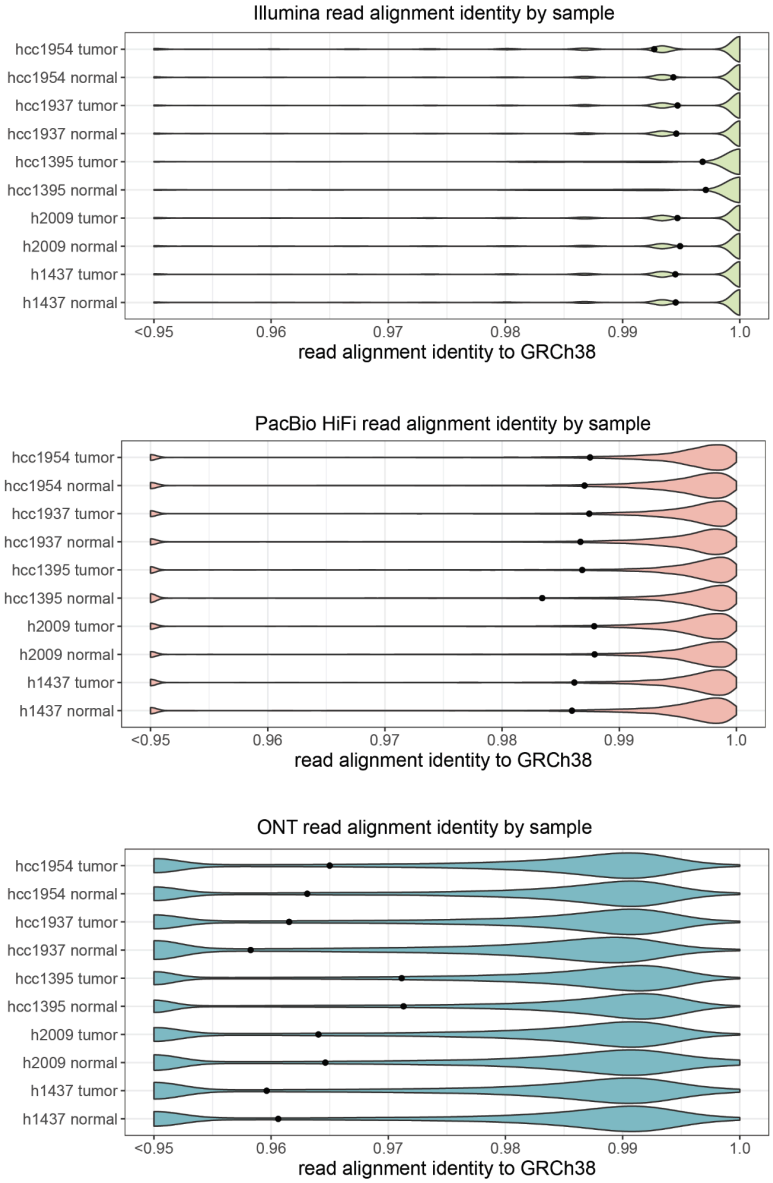
Supplementary Figure 3: indel classification counts for SEQC2 benchmark and each benchmark set derived from DeepSomatic HCC1395 models **(left)** and multi-cancer models **(right)** in high confidence regions. Plots were generated using SigProfileMatrixGenerator.



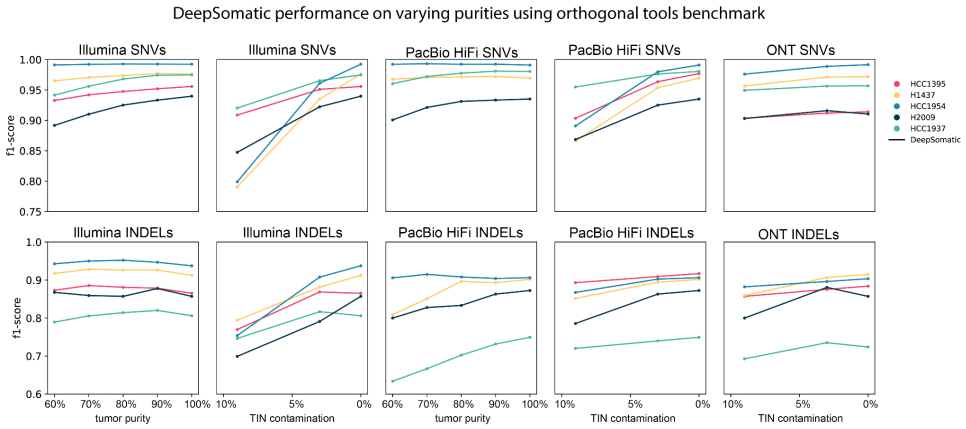
Supplementary Figure 4: Performance of DeepSomatic HCC1395 model vs multi-cancer model against orthogonal tools benchmark on the whole genome.



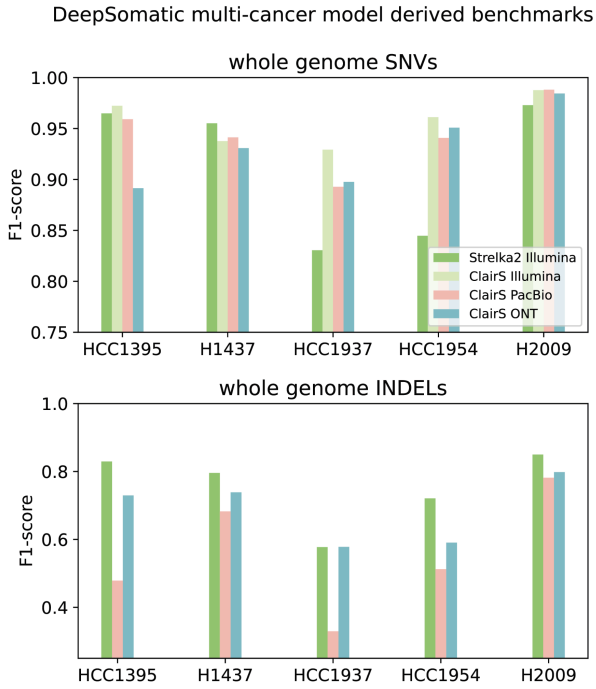
Supplementary Figure 5: Performance of somatic variant callers against the training set derived from DeepSomatic HCC1395 models.



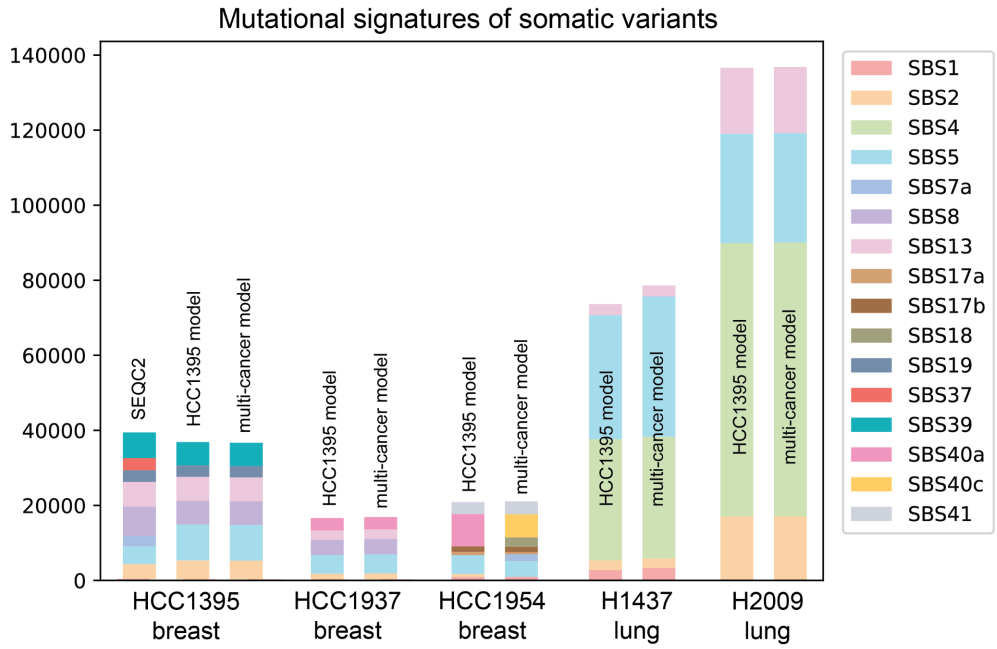
Supplementary Figure 6: Read alignment identity to GRCh38 by sequencing technology and cell line sample. Points represent the mean for each cell line sample.



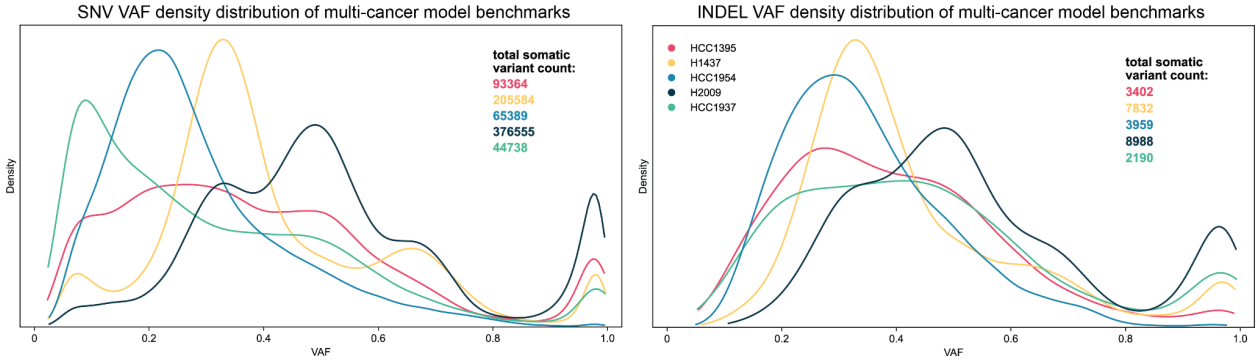
Supplementary Figure 7: Performance of DeepSomatic multi-cancer model on various tumor purities and TIN contamination against orthogonal tools benchmark on chromosome 1.



Supplementary Figure 8: Benchmarking somatic variant callers using DeepSomatic multi-cancer model derived benchmarks.



Supplementary Figure 9: Comparison of mutational signatures of somatic variants called by DeepSomatic HCC1395 models and multi-cancer models as well as the SEQC2 benchmark, showing them to be highly consistent.



Supplementary Figure 10: Variant allele frequency (VAF) distributions of somatic variant sets derived using DeepSomatic multi-cancer models in high confidence regions, represented as Kernel Density Estimate (KDE) plots.

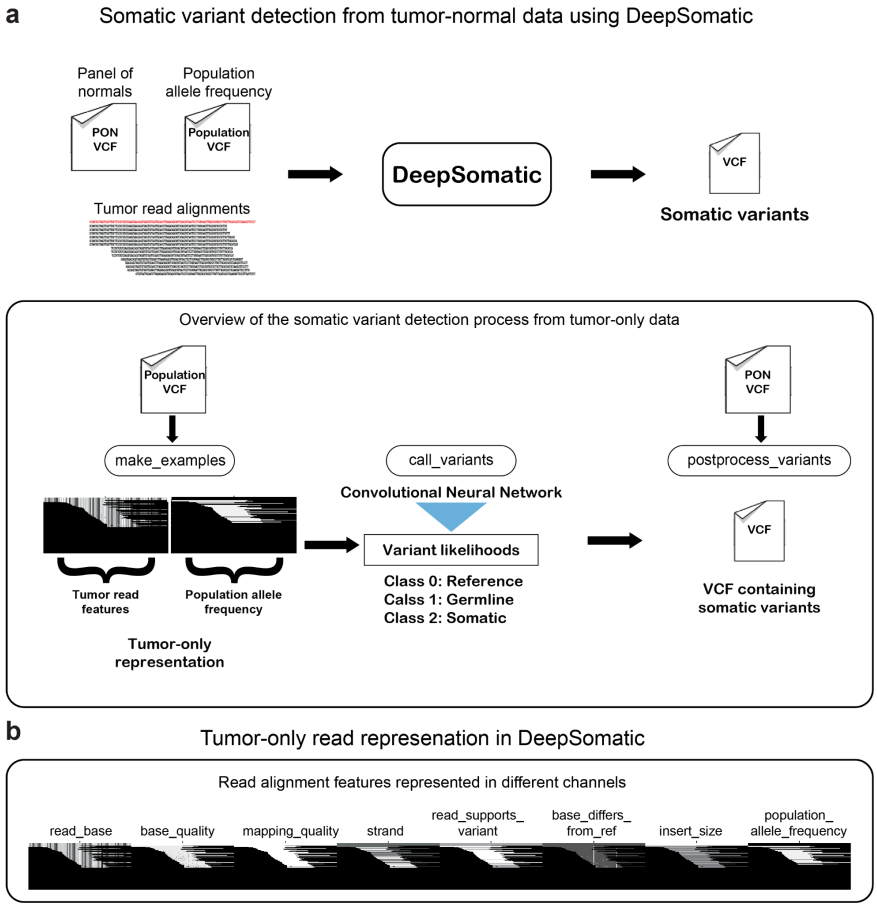


Figure 11: DeepSomatic tumor-only variant calling overview

(a) Overview of DeepSomatic tumor-only mode. **(b)** Tumor-only read alignment features represented including population allele frequency.

Command lines

Read Alignment

Minimap2 (v2.26)

https://github.com/lh3/minimap2

```
Unset
minimap2 -ax map-ont \
        -t ~{minimap_threads} \
        -k 17 -y -K 5G --eqx \
        ~{reference} \
        ~{fastq} \
        | samtools sort -@ ~{samtools_threads} -m 4G > ~{temp_prefix}_GRCh38.sorted.bam
```

Pbmm2 (docker quay.io/biocontainers/pbmm2:1.13.1--h9ee0642_0)

https://github.com/PacificBiosciences/pbmm2

```
Unset
echo ~{ubam1} > myfiles.fofn
        echo ~{ubam2} >> myfiles.fofn

pbmm2 align ~{reference} myfiles.fofn ~{sample_name}_HiFi.GRCh38.sorted.bam \
        --sort \
        --min-length 50 \
        --sample ~{sample_name} \
        --preset HiFi \
        --num-threads ~{threads} \
        -J 4 \
        --log-level INFO
```

Bwa-mem2 (v2.2.1)

https://github.com/bwa-mem2/bwa-mem2

```
Unset
# index reference
bwa-mem2 index ~{reference}

# align with bwa-mem2 and sort with samtools
bwa-mem2 mem -M -t ~{threads} ~{reference} ~{fastq1} ~{fastq2} \
| samtools sort -@4 -m 4G > ~{sample_name}_Illumina.GRCh38.sorted.bam

# index bam
samtools index -@ ~{threads} ~{sample_name}_Illumina.GRCh38.sorted.bam
```

Calculating depth of BAMs (samtools v1.13)

```
Unset
samtools depth ${bam} | awk '{sum+=$3} END { print "Average = ",sum/NR}'
```

Calculating alignment identity and N50 (wambam)
https://github.com/nanoporegenomics/wambam

```
Unset
time docker run --rm -i -u `id -u`:`id -g` \
  -v ${working_path}:${working_path} \
    quay.io/jmonlong/wambam \
    wam -i ${NORMAL_BAM} -o wambam_results_normal

time docker run --rm -i -u `id -u`:`id -g` \
  -v ${working_path}:${working_path} \
    quay.io/jmonlong/wambam \
    wam -i ${TUMOR_BAM} -o wambam_results_tumor
```

Variant Calling

DeepSomatic

https://github.com/google/deepsomatic

```
Unset
# model types:
<WGS|WES|PACBIO|ONT|FFPE_WGS|FFPE_WES|WGS_TUMOR_ONLY|PACBIO_TUMOR_ONLY|ONT_TUMOR_ONLY>

time docker run --rm -i -u `id -u`:`id -g` \
        -v ${working_path}:${working_path} \
        google/deepsomatic:v17_rc1_08082024 \
        run_deepsomatic \
        --model_type=${MODEL} \
        --ref=${REFERENCE} \
        --reads_normal=${NORMAL_BAM} \
        --reads_tumor=${TUMOR_BAM} \
        --output_vcf=${OUTPUT_DIR}/${output_prefix}.vcf.gz \
        --sample_name_tumor="${sample}_WGS_tumor" \
        --sample_name_normal="${sample}_WGS_normal" \
        --num_shards=${THREADS} \
        --logging_dir=${OUTPUT_DIR}/${sample}/${log_prefix}/logs \
        --regions=chr1
```

ClairS (v0.2.0)

https://github.com/HKU-BAL/ClairS

```
Unset
  time docker run --rm -i -u `id -u`:`id -g` \
          -v ${working_path}:${working_path} \
          hkubal/clairs:v0.2.0 \
          /opt/bin/run_clairs \
          --tumor_bam_fn ${TUMOR_BAM} \
          --normal_bam_fn ${NORMAL_BAM} \
          --ref_fn ${REF} \
          --threads ${THREADS} \
          --platform ${PLATFORM} \
          --output_dir ${OUTPUT_DIR} \
          --sample_name "ClairS" \
          --region "chr1:1-248956422"
```

Strelka2 (v2.9.10)
https://github.com/Illumina/strelka

```
Unset
  ${STRELKA_INSTALL_PATH}/configureStrelkaSomaticWorkflow.py \
  --normalBam ${NORMAL_BAM} \
  --tumorBam ${TUMOR_BAM} \
  --referenceFasta ${REF} \
  --runDir ${STRELKA_ANALYSIS_PATH} \
  --callRegions ${BED}

  ${STRELKA_ANALYSIS_PATH}/runWorkflow.py -m local -j 64
```

DeepVariant HYBRID_PACBIO_ILLUMINA (v1.6.1)
https://github.com/google/deepvariant

```
Unset
time docker run --rm -i -u `id -u`:`id -g` \
  -v ${working_path}:${working_path} \
  google/deepvariant:1.6.1 \
  /opt/deepvariant/bin/run_deepvariant \
  --model_type=HYBRID_PACBIO_ILLUMINA \
  --ref=${REFERENCE} \
  --reads=${HYBRID_BAM} \
  --output_vcf=${OUTPUT_VCF} \
  --num_shards=${THREADS} \
  --logging_dir="logs"
```

Generating training sets
https://github.com/jimin001/DeepSomatic_manuscript/blob/main/vcf_intersection_complex_v2.py

```
Unset
# 1. merge Illumina, PacBio, ONT variants
bcftools merge --force-samples --threads 16 ${ILLUMINA} ${HIFI} ${ONT} | bgzip >
merge.vcf.gz
bcftools index -t merge.vcf.gz

# 2. filter out variants that do not meet our criteria for "high confident" variant
python3 vcf_intersection_complex_v2.py -v merge.vcf.gz -i merge.vcf.gz.tbi -f 'filter4' -o
filter.vcf.gz
bcftools index -t filter.vcf.gz

# 3. filter for 'PASS' (somatic) variants only
bcftools filter -i 'FILTER="PASS"' filter.vcf.gz | bgzip > filter_somaticOnly.vcf.gz
bcftools index -t filter_somaticOnly.vcf.gz

# 4. merge germline variants to filter.vcf.gz
bcftools merge --force-samples --threads 4 filter_somaticOnly.vcf.gz germline.vcf.gz >
truth.vcf.gz
```

Generating high confidence region BED files

https://github.com/jimin001/DeepSomatic_manuscript/blob/main/vcf_to_bed_v4.py

```
Unset
# 1. convert merge.vcf.gz to BED, filter_somaticOnly.vcf.gz to BED
python3 vcf_to_bed_v4.py -v merge.vcf.gz -i merge.vcf.gz.tbi -t 'deepsomatic' -o merge.bed
python3 vcf_to_bed_v4.py -v filter_somaticOnly.vcf.gz -i filter_somaticOnly.vcf.gz.tbi -t
'deepsomatic' -o filter_somaticOnly.bed

# 2. obtain confusing sites (variants called in merge.vcf but not in filter.vcf)
bedtools subtract -a merge.bed -b filter_somaticOnly.bed > confusing_sites.bed

# 3. remove confusing sites
bedtools subtract -a callableregions.bed -b confusing_sites.bed > highconfidence.bed

# 4. convert sv.vcf.gz generated with Severus to BED
python3 vcf_to_bed_v4.py -v sv.tumor.vcf.gz -i sv.tumor.vcf.gz,tbi -t 'severus' -o
sv.tumor.bed -b 1000
python3 vcf_to_bed_v4.py -v sv.normal.vcf.gz -i sv.normal.vcf.gz,tbi -t 'severus' -o
sv.normal.bed -b 1000

# 5. remove SV sites
bedtools subtract -a highconfidence.bed -b sv.tumor.bed | bedtools subtract -a stdin -b
sv.normal.bed > highconfidence_minusSVs.bed
```

Generating callable regions

Unset

```
# 1. CallableLoci
# minBaseQuality=7 for ONT reads, minBaseQuality=20 for PacBio/Illumina reads
time docker run -i -u `id -u`:`id -g` \
-v ${working_path}:${working_path} \
broadinstitute/gatk3:3.8-1 \
java -Xmx8g -jar GenomeAnalysisTK.jar \
-T CallableLoci \
-R GRCh38.d1.vd1.fa \
-I ${BAM} \
--maxDepth ${DEPTH} \
--maxFractionOfReadsWithLowMAPQ 0.1 \
--maxLowMAPQ 1 \
--minBaseQuality ${minBaseQuality} \
--minMappingQuality 20 \
--minDepth 10 \
--minDepthForLowMAPQ 10 \
--summary ${OUTPUT_DIR}/${output_prefix}.summary \
-o ${OUTPUT_DIR}/${output_prefix}.callable.bed \
--filter_reads_with_N_cigar \
--filter_mismatching_base_and_quals \
--filter_bases_not_stored \
--allow_potentially_misencoded_quality_scores

# 2. filter for callable regions in tumor and normal beds
grep "CALLABLE" ${output_prefix}.T.callable.bed > ${output_prefix}.T.callable.only.bed
grep "CALLABLE" ${output_prefix}.N.callable.bed > ${output_prefix}.N.callable.only.bed

# 3. filter for regions callabe by both tumor and normal beds
docker run --rm -i -u `id -u`:`id -g` \
-v ${working_path}:${working_path} \
lethalfang/somaticseq:2.7.2 \
/opt/somaticseq/utilities/lociCounterWithLabels.py \
-fai /private/groups/patenlab/jimin/data/reference/GRCh38.d1.vd1.fa.fai \
-beds ${output_prefix}.T.callable.only.bed \
      ${output_prefix}.N.callable.only.bed \
-labels ${output_prefix}_T ${output_prefix}_N | awk -F '\t' '$4>=2' >
${output_prefix}_N_and_T_MajorityCallable.bed

# 4. merge neighboring positions
bedtools merge -i ${output_prefix}_N_and_T_MajorityCallable.bed >
${output_prefix}_N_and_T_MajorityCallable.merge.bed

# 5. do above steps for Illumina, PacBio, ONT and concatenate callable regions together
cat ${output_prefix}_Illumina_N_and_T_MajorityCallable.merge.bed
${output_prefix}_PacBio_N_and_T_MajorityCallable.merge.bed
${output_prefix}_ONT_N_and_T_MajorityCallable.merge.bed >
${sample}_MajorityCallable.merge.bed
```

Generating titration bams

https://github.com/jimin001/DeepSomatic_manuscript/blob/main/split_bam_tumor.sh
https://github.com/jimin001/DeepSomatic_manuscript/blob/main/split_bam_normal.sh
https://github.com/jimin001/DeepSomatic_manuscript/blob/main/tumor_purity_titration.sh
https://github.com/jimin001/DeepSomatic_manuscript/blob/main/normal_purity_titration.sh

```
Unset
# split tumor bam to spike in for purity titrations
split_bam_tumor.sh \
-n ${tumor_bam} \
-q ${tumor_coverage} \
-g ${tumor_goal_coverage} \
-e ${tumor_evaluation_bam} \
-p ${platform} \
-s ${sample} \
-o ${output_directory}

# split normal bam to spike in for purity titrations
split_bam_normal.sh \
-n ${normal_bam} \
-q ${normal_coverage} \
-g ${normal_goal_coverage} \
-e ${normal_evaluation_bam} \
-p ${platform} \
-s ${sample} \
-o ${output_directory}

# tumor purity bams
tumor_purity_titration.sh \
-t ${tumor_bam} \
-c ${tumor_coverage} \
-n ${normal_bam} \
-q ${normal_coverage} \
-g ${tumor_goal_total_coverage} \
-p ${platform} \
-s ${sample} \
-o ${output_directory}

# normal purity bams
normal_purity_titration.sh \
-t ${tumor_bam} \
-c ${tumor_coverage} \
-n ${normal_bam} \
-q ${normal_coverage} \
-x ${normal_goal_total_coverage} \
-p ${platform} \
-s ${sample} \
-o ${output_directory}
```

Mutational signature analysis

https://github.com/AlexandrovLab/SigProfilerAssignment

```
Unset
# multi-cancer version
import SigProfilerAssignment as spa
from SigProfilerAssignment import Analyzer as Analyze
Analyze.cosmic_fit(samples="input_vcfs",
                   output="outputs",
                   input_type="vcf",
                   context_type="96",
                   genome_build="GRCh38",
                   cosmic_version=3.4, make_plots=True, verbose=True)

# 1395 version
import SigProfilerAssignment as spa
from SigProfilerAssignment import Analyzer as Analyze
Analyze.cosmic_fit(samples="input_vcfs",
                   output="outputs",
                   input_type="vcf",
                   context_type="96",
                   genome_build="GRCh38",
                   cosmic_version=3.4, make_plots=True, verbose=True)
```

Mutational matrix generation

https://github.com/AlexandrovLab/SigProfilerMatrixGenerator

```
Unset
from SigProfilerMatrixGenerator import install as genInstall
genInstall.install('GRCh38', rsync=False, bash=True)

# HCC1395 model
python3
from SigProfilerMatrixGenerator.scripts import SigProfilerMatrixGeneratorFunc as matGen
matrices = matGen.SigProfilerMatrixGeneratorFunc("HCC1395_model", "GRCh38",
"input_vcfs",plot=True, exome=False, bed_file=None, chrom_based=False, \
    tsb_stat=False, seqInfo=False, cushion=100)

# multi-cancer model
python3
from SigProfilerMatrixGenerator.scripts import SigProfilerMatrixGeneratorFunc as matGen
matrices = matGen.SigProfilerMatrixGeneratorFunc("multi-cancer_model", "GRCh38",
"input_vcfs",plot=True, exome=False, bed_file=None, chrom_based=False, \
  tsb_stat=False, seqInfo=False, cushion=100)
```

Benchmarking

Som.py

https://github.com/Illumina/hap.py/blob/master/doc/sompy.md

```
Unset
  time docker run --rm -it \
    -v ${working_path}:${working_path} \
    pkrusche/hap.py:latest \
    /opt/hap.py/bin/som.py \
    -N \
    ${TRUTH} \
    ${QUERY} \
    --restrict-regions ${BED} \
    -r ${REFERENCE} \
    -o "sompy" \
    --feature-table generic \
    -l chr1
```