Virdetect Manual

Contents

1. Getting Started

The files needed for the virdetect workflow can be downloaded from GitHub at https://github.com/dmarron/virdetect

The workflow can either be run as several command line steps (Section 3) or from cwl files with a cwl runner (Section 4).

Before the workflow can be run, the appropriate genome files (.fa) will need to be downloaded and then indexed with the STAR aligner (https://github.com/alexdobin/STAR). For human data, download hg38_noEBV.fa and virus_masked_hg38.fa from the virdetect gitHub. For mouse data, download mm10.fa from UCSC or ensembl and download virus_masked_mm10.fa from the virdetect github. For the commands to index this genomes with STAR, see Section 2.

2. Index Genomes with STAR

To index the genome .fa files, download the STAR aligner and the necessary files for the human and mouse workflows.

For human data, download hg38_noEBV.fa and a hg38 gtf file (Example - ftp://ftp.ensembl.org/pub/release-89/gtf/homo_sapiens).

For mouse data, download mm10.fa and a mm10 gtf file (Example - ftp://ftp.ensembl.org/pub/release-89/gtf/mus_musculus).

Then run the following commands to index the main genome and virus genomes:

For human data:

```
STAR --runThreadN 8 --genomeChrBinNbits 14 --runMode
genomeGenerate --genomeDir hg38_star_dir --genomeFastaFiles hg38.fa
--sjdbGTFfile hg38_gtf.gtf
STAR --runThreadN 1 --runMode genomeGenerate
--genomeSAindexNbases 7 --genomeDir hg38_virus_dir
--genomeFastaFiles virus_masked_hg38.fa
```

For mouse data:

```
STAR --runThreadN 8 --genomeChrBinNbits 14 --runMode
genomeGenerate --genomeDir mm10_star_dir --genomeFastaFiles
mm10.fa --sjdbGTFfile mm10_gtf.gtf
STAR --runThreadN 1 --runMode genomeGenerate
--genomeSAindexNbases 7 --genomeDir mm10_virus_dir
--genomeFastaFiles virus_masked_mm10.fa
```

3. Run workflow with command line

After the genomes are indexed with STAR, the virdetect workflow is ready to be run. To run virdetect from the command line, download all java files, sh files, and jar files from the virdetect github. The files virdetect_hg38_PE.lbgcluster.inc and virdetect_mm10_PE.lbgcluster.inc detail the steps workflow steps for human and mouse data respectively. The workflow steps are listed below as well. If a line has cpus:x and mem:y as a comment, that indicates to submit that job with x threads and y gigs of memory requested.

For human data the workflow is:

```
perl mergeFastqs.pl -f $fq1,$fq2,$fq3,$fq4,$fq5,$fq6,$fq7,$fq8 -o
$outdir/working/temp # cpus:9

STAR --runThreadN 16 --genomeDir hg38_star_dir --readFilesIn
$outdir/working/temp_R1.fastq $outdir/working/temp_R2.fastq
--outFilterMultimapNmax 1000 --outSAMunmapped Within
--outFileNamePrefix $outdir/working/STAR_ #cpus:16 mem:2

sh awk_column3_star.sh $outdir/working/STAR_Aligned.out.sam >
$outdir/working/unaligned.sam

sh /home/dmarron/workspace/scripts/awk_unalignedfq_1.sh
$outdir/working/unaligned.sam > $outdir/working/unaligned_1.fastq && sh
```

```
awk_unalignedfq_2.sh $outdir/working/unaligned.sam >
$outdir/working/unaligned_2.fastq
```

```
STAR --genomeDir hg38_virus_dir --readFilesIn
$outdir/working/unaligned_1.fastq $outdir/working/unaligned_2.fastq
--runThreadN 16 --outFilterMismatchNmax 4 --outFilterMultimapNmax 1000
--limitOutSAMoneReadBytes 1000000 --outFileNamePrefix
$outdir/STAR_virus_ # cpus:16 mem:2
```

```
java -Xmx4G -cp picard-1.92.jar:sam-1.92.jar:countStarViralAlignments
$sample_name $outdir/STAR_virus_Aligned.out.sam
$outdir/viralReadCounts.txt #mem:8
```

```
mv $outdir/working/STAR_Log* $outdir/ && ls -l $outdir/working/ >
$outdir/output/fileSizes.txt
```

```
rm -rfv $outdir/working/* # cleanup
```

For mouse data the workflow is:

```
perl mergeFastqs.pl -f $fq1,$fq2,$fq3,$fq4,$fq5,$fq6,$fq7,$fq8 -o
$outdir/working/temp # cpus:9
```

```
STAR --runThreadN 16 --genomeDir mm10_star_dir --readFilesIn
$outdir/working/temp_R1.fastq $outdir/working/temp_R2.fastq
--outFilterMultimapNmax 1000 --outSAMunmapped Within
--outFileNamePrefix $outdir/working/STAR_ #cpus:16 mem:2
```

```
sh awk_column3_star.sh $outdir/working/STAR_Aligned.out.sam >
$outdir/working/unaligned.sam
```

```
sh /home/dmarron/workspace/scripts/awk_unalignedfq_1.sh
$outdir/working/unaligned.sam > $outdir/working/unaligned_1.fastq && sh
```

```
awk_unalignedfq_2.sh $outdir/working/unaligned.sam >
$outdir/working/unaligned_2.fastq
```

```
STAR --genomeDir mm10_virus_dir --readFilesIn
$outdir/working/unaligned_1.fastq $outdir/working/unaligned_2.fastq
--runThreadN 16 --outFilterMismatchNmax 4 --outFilterMultimapNmax 1000
--limitOutSAMoneReadBytes 1000000 --outFileNamePrefix
$outdir/STAR_virus_ # cpus:16 mem:2
```

```
java -Xmx4G -cp picard-1.92.jar:sam-1.92.jar:countStarViralAlignments
$sample_name $outdir/STAR_virus_Aligned.out.sam
$outdir/viralReadCounts.txt #mem:8
```

```
mv $outdir/working/STAR_Log* $outdir/ && ls -l $outdir/working/ >
$outdir/output/fileSizes.txt
```

```
rm -rfv $outdir/working/* # cleanup
```

4. Run workflow with cwl

To run virdetect with a cwl runner instead of from the command line, download the cwl files, jar files, java files, and sh files from the virdetect github. Then make a .yml file that contains the input fastqs for the workflow.

For human data the yml file will look like:

```
 - class: File
     path: R1.fastq
       - class: File
           path: R2.fastq
           referenceGenome:
             class: Directory
               size: 30000
                 path: hg38_star_dir
```

```
         viralReferenceGenome:
          class: Directory
           size: 30000
            path: hg38_virus_dir
            sampleName: sample_name
```

For mouse data the yml file will look like:

```
 - class: File
   path: R1.fastq
    - class: File
      path: R2.fastq
      referenceGenome:
       class: Directory
        size: 30000
         path: mm10_star_dir
         viralReferenceGenome:
          class: Directory
           size: 30000
            path: mm10_virus_dir
            sampleName: sample_name
```

The run the cwl with the command:

```
cwl-runner --submit Virdetect.cwl yml_file.yml
```

5. Visualization

After running virdetect, it may be of interest to visualize coverage of specific viruses. To run the visualization, download makeRTable.java, plotTable.R, and the jar files from the virdetect github. The run the commands:

```
java -Xmx4G -cp picard-1.92.jar:sam-1.92.jar:makeRTable virus_name STAR_virus.Aligned.out.bam viralRTable.txt
Rscript plotTable.R viralRTable.txt
```

Those commands will plot the coverage of the virus strain given by the parameter virus_name.

6. Masking custom genomes

It may be of interest to run virdetect with custom virus strains rather than the ones provided in virus_masked_hg38.fa and virus_masked_mm10.fa. To do so, download simulateReads.java, makeAlignedBed.java and maskGenome.java from the virdetect github. To mask your custom genome fa (custom_virus.fa) for human data, run the following commands:

```
java simulateReads custom_virus.fa sim.fastq
STAR --runThreadN 16 --genomeDir mm10_star_dir --readFilesIn sim.fastq
--outFilterMismatchNmax 5 --outFilterMultimapNmax 1080
--outFileNamePrefix $outdir/STAR_ #cpus:16 mem:2
java makeAlignedBed STAR_Aligned.out.sam aligned.bed
java maskGenome custom_virus.fa aligned.bed custom_masked_virus.fa
```

The resulting file, custom_masked_virus.fa, contains the masked genome that can then be indexed with STAR (with command from section 2) and used with virdetect.