Deep Variant Work Flow

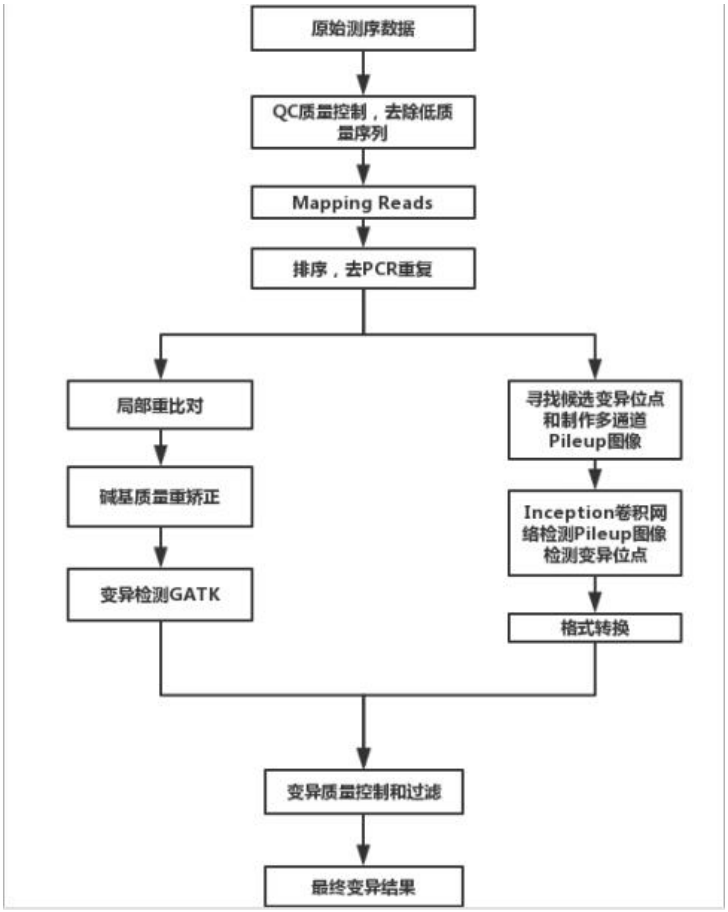
Linux 台式机账号密码：zoubohao , XJYzbh19960919 台式机已经安装好环境，可以直接运行流程

1. Install

Github Link : https://github.com/google/deepvariant

Github quick install from binary :
https://github.com/google/deepvariant/blob/r0.6/docs/deepvariant-quick-start.md
Google 自动安装对应的依赖和环境。

2. Usage



Note :

- 必须对参考基因组进行索引 index ,
- 对待测样本进行质量控制，Mapping , 去重复等操作。最后在同文件夹中必须含有

bam 文件和对应的 bai 文件 。

Note:以下所有包的使用都是按照流程顺序一步一步使用的，一般来说，某一步的输入是上一步的输出文件。

####### 预处理：

1. # Trimmomatic 去除测序质量低的序列和接头序列（分 PE（pair end ）和 SE(single end) 模式）-phred33 -phred64 代表测序质量，现在一般用 33：
   PE :
   java -jar trimmomatic.jar PE -phred33 \
   ../WGS_Work_Flow/ReadsData/R1.fastq \ **## 输入**
   ../WGS_Work_Flow/ReadsData/R2.fastq \ **## 输入**
   ../WGS_Work_Flow/ReadsData/outR1.fastq.gz \ **## 输出**
   ../WGS_Work_Flow/ReadsData/outR1Trimm.fastq.gz \ **## 输出低质量**
   ../WGS_Work_Flow/ReadsData/outR2.fastq.gz \ **## 输出**
   ../WGS_Work_Flow/ReadsData/outR2Trimm.fastq.gz \ **## 输出低质量**
   ILLUMINACLIP:adapters/TruSeq3-PE.fa:2:30:10 SLIDINGWINDOW:5:20 LEADING:5 TRAILING:5 MINLEN:50

2. # 建立人类参考基因组索引
   bwa index HumenChromos.fasta

3. # 将经过质量控制的 reads 映射到基因组上并输出 BAM 文件
   bwa mem -t 4 -R '@RG\tID:sra_data\tPL:illumina\tLB:Non\tSM:humen' \ ##注意 ID,LB,SM
   ~/WGS_Work_Flow/ChromeIndexFiles/GRCh38.fna \ ##人类参考基因组文件
   ~/WGS_Work_Flow/ReadsData/sra_data.fastq **## 输入**
   | samtools view -S -b - > Humen.bam **## 输出**并转换为 bam 文件

4. # 将 BAM 文件排序
   samtools sort Humen.bam Humen_sort.bam

5. # 删除重复序列
   Samtools rmdup -S (PE model) Humen_sort.bam **## 输入**
   Humen_sort_markdup.bam **## 输出**
   或者
   java -jar picard.jar MarkDuplicates \ ##删除重复序列命令
   REMOVE_DUPLICATES=true \ ## 直接在文件中移除重复序列
   I=ReadsData/458QcMappingSort.bam \ ## **输入**数据
   O=ReadsData/458QMSD.bam \ ## **输出**数据
   M=ReadsData/458QMSDMeticx.txt ## 中间生成的矩阵

6. # 对最终 BAM 文件建立索引
   samtools index Humen_sort_markdup.bam

7. # 为人类参考基因组 fasta 文件建立 fai 索引
   samtools faidx data.fasta

####### Deep Variant 检测

第一步、寻找变异位点

```
# 建立图片
python bin/make_examples.zip \
  --mode calling \          ## 模式的选择
  --ref "${REF}" \          ## 参照基因序列的 fasta 文件
  --reads "${BAM}" \        ## 输入处理之后的 BAM 文件
  --examples "${OUTPUT_DIR}/examples.tfrecord.gz"       ## 输出文件以及格式
```

第二步、检测

```
# 检测变异
python bin/call_variants.zip \
 --outfile "${CALL_VARIANTS_OUTPUT}.tfrecord.gz" \   ## 输出文件以及格式
 --examples "${OUTPUT_DIR}/examples.tfrecord.gz" \    ## 输入上一步输出的文件
 --checkpoint "${MODEL}"          ##   tensroflow 对应的模型参数
```

第三步、后处理

```
# 格式转换
python bin/postprocess_variants.zip \
  --ref "${REF}" \       ##参照基因序列的 fasta 文件
  --infile "${CALL_VARIANTS_OUTPUT}" \     ##输入上一步输出的文件
--outfile "${FINAL_OUTPUT_VCF}.vcf.gz"       ##最终输出 VCF 文件
```

####### 优缺点：

1. DeepVariant 模型参数固定，无法使用自己的数据去训练。
2. 寻找变异位点的步骤（第一步）时间耗费很长。
3. 使用深度学习技术检测 SNV 位点，在一定程度上提升了准确度。
4. 没有 BUG。

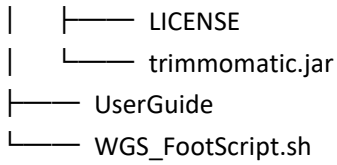
####### 评估：

1. 已经将跑出来的 8 个样本结果交给鑫凯进行评估。
2.

####### 数据路径与样式

TestFootScript/

```
├──── DeepVariant-Binary        ## python 的二进制包文件夹
│    ├──── call_variants.zip
│    ├──── licenses.zip
│    ├──── make_examples.zip
│    ├──── model_eval.zip
│    ├──── model_train.zip
│    ├──── postprocess_variants.zip
│    ├──── run-prereq.sh
│    └──── settings.sh
├──── DeepVariant-Model        ##TensorFlow 模型参数存储文件
│    ├──── model.ckpt.data-00000-of-00001
│    ├──── model.ckpt.index
│    └──── model.ckpt.meta
├──── DeepVariantWork.sh
├──── hg19
│    ├──── hg19.fasta
│    ├──── hg19.fasta.amb
│    ├──── hg19.fasta.ann
│    ├──── hg19.fasta.bwt
│    ├──── hg19.fasta.fai
│    ├──── hg19.fasta.pac
│    └──── hg19.fasta.sa
├──── hg38T
│    ├──── GRCh38.fasta
│    ├──── GRCh38.fasta.amb
│    ├──── GRCh38.fasta.ann
│    ├──── GRCh38.fasta.bwt
│    ├──── GRCh38.fasta.fai
│    ├──── GRCh38.fasta.pac
│    ├──── GRCh38.fasta.sa
│    └──── GRCh38.pac
├──── PATH
├──── picard.jar
├──── ReadsData             ##  测试数据文件夹
│    ├──── Sample_1.fastq
│    └──── Sample_2.fastq
├──── Trimmomatic
│    ├──── adapters
│    │    ├──── NexteraPE-PE.fa
│    │    ├──── TruSeq2-PE.fa
│    │    ├──── TruSeq2-SE.fa
│    │    ├──── TruSeq3-PE-2.fa
│    │    ├──── TruSeq3-PE.fa
│    │    └──── TruSeq3-SE.fa
```


├── LICENSE
└── trimmomatic.jar
├── UserGuide
└── WGS_FootScript.sh