Getting started with VirSorter2 V.2

Jiarong Guo[1]

[1]Ohio State University, Columbus

Version 2

Jul 09, 2020

1 Works for me dx.doi.org/10.17504/protocols.io.bidpka5n

iVirus

Jiarong Guo

DOI

dx.doi.org/10.17504/protocols.io.bidpka5n

PROTOCOL CITATION

Jiarong Guo 2020. Getting started with VirSorter2 . **protocols.io** https://dx.doi.org/10.17504/protocols.io.bidpka5n

CREATED

Jul 09, 2020

LAST MODIFIED

Jul 09, 2020

PROTOCOL INTEGER ID

39055

BEFORE STARTING

VirSorter2 applies a multi-classifier, expert-guided approach to detect diverse DNA and RNA virus genomes. It has made major updates to its previous version:

- work with more viral groups including dsDNAphage, ssDNA, RNA, NCLDV (Nucleocytoviricota), *lavidaviridae*
- apply machine learning to estimate viralness using genomic and taxonomic features and hallmark gene counts
- train with high quality virus genomes from metagenomes or other sources

This tutorial assumes you have access to an linux machine, and the command lines in this tutorial works in "terminal" app of linux machine. We will cover four sub-commands 1) "virsorter setup" (download database), 2) "virsorter run" (the main viral identification pipeline), 3) "virsorter train-feature" (extract feature from viral genomes for training model), and 4) "virsorter train-model" (train classifier model).

Installation

1

Skip this step if you already have VirSorter2 installed. You can find out by type in the following command in terminal:

```
virsorter -h
```

If you see message like below, then you have VirSorter2 installed already.

```
Usage: virsorter [OPTIONS] COMMAND [ARGS]...
```

```
  virsorter - workflow for identifying viral sequences

Options:
  --version   Show the version and exit.
  -h, --help  Show this message and exit.

Commands:
  run            run virsorter main workflow
  setup          download reference files (~10GB) and install dependencies
  train-feature  subcommand for training feature of customized classifier
  train-model    subcommand for training customized classifier model
```

A message like this means VirSorter2 is not installed.

```
-bash: virsorter: command not found
```

Option 1:

Conda is the easiest way to install VirSorter2. Conda can install by following this link.

```
conda intall -c bioconda virsorter
```

Option 2:

To install the development version (most updated but may not work all the time):

```
conda create -n vs2 python=3 scikit-learn=0.22.1 imbalanced-learn pandas seaborn hmmer
prodigal screed ruamel.yaml snakemake=5.16.0 click
conda activate vs2
git clone https://github.com/jiarong/VirSorter2.git
cd VirSorter2
pip install -e .
```

Download database and dependencies

2 Before running VirSorter2, users must download all databases and install dependencies (takes 10+ mins, but this only need to be done once). The following command line downloads databases and dependencies to "db" directory, and its location is recorded in the tool configuration as a default, so you do not need to type "--db-dir" for other VirSorter2 subcommands.

```
virsorter setup -d db -j 4
```

Quick run

3 To run viral sequence identification:

```
# fetch testing data
wget -O test.fa https://raw.githubusercontent.com/jiarong/VirSorter2/master/test/8seq.fa

# run classification with 4 threads (-j) and test-out as output diretory (-w) virsorter run
-w test.out -i test.fa -j 4

# check output
ls test.out
```

Due to large HMM database that VirSorter2 uses, this small dataset takes a few mins to finish. In the output directory

Citation: Jiarong Guo (07/09/2020). Getting started with VirSorter2 . https://dx.doi.org/10.17504/protocols.io.bidpka5n

(test.out), three files are useful:

- final-viral-combined.fa: identified viral sequences
- final-viral-score.tsv: table with score of each viral sequences across groups
- final-viral-boundary.tsv: table with boundary information

More details of output files can be found in Section "Detailed description on output files" below.

NOTE
Note that suffix "||full" or "||{i}index_partial" have been added to original sequence names to differentiate sub-sequences in case of multiple viral subsequences found in one contig ("{i}" can be numbers starting from 0 to max number of viral fragments found in that contig).

More options

4 **Choosing viral groups ("--include-groups")**

VirSorter2 finds all viral groups currently included (ssDNAphage, NCLDV , RNA, ssDNA virus, and *lavidavirida*) by default. You can use--include-groupsto chose specific groups:

```
rm -rf test.out
virsorter run -w test.out -i test.fa --include-groups "dsDNAphage,ssDNA" -j 4
```

Re-run with different score cutoff ("--min-score")

VirSorter2 takes one positional argument, "all" or "classify". The default is all, which means running the whole pipeline, including 1) preprocessing, 2) annotation (feature extraction), and 3) classification. The main computational bottleneck is the annotation step, taking about 95% of CPU time. In case you just want to re-run with different score cutoff (--min-score), the "classify" argument can skip the annotation steps, and only re-run classify step.

```
virsorter run -w test.out -i test.fa --include-groups "dsDNAphage,ssDNA" -j 4 --min-score
0.8 classify
```

Speed up a run (--provirus-off)

In case you need to have some results quickly, there are two options: 1) turn off provirus step with "--provirus-off"; this reduces sensitivity on sequences that are only partially virus; 2) subsample ORFs (Open Reading Frame) from each sequence with "--max-orf-per-seq"; This option subsamples ORFs to a cutoff if a sequence has more ORFs than that. Note that this option is only availale when--provirus-offis used.

```
rm -rf test.out
virsorter run -w test.out -i test.fa --provirus-off --max-orf-per-seq 20
```

Other options

You can "runvirsorter run -h" to see all options. VirSorter2 is a wrapper around snakemake, a great pipeline management tool designed for reproducibility, and running on computer clusters. All snakemake options still works here. You just need to append those snakemake option to virsorter options (after the "all" or "classify" argument). For example, the "--forceall" snakemake option can be used to re-run the pipeline.

```
virsorter run -w test.out -i test.fa --provirus-off --max-orf-per-seq 20 --forceall
```

NOTE:
When you re-run any VirSorter2 command, it will pick up at the step (rule in snakemake term) where it stopped last time. It will do nothing if it suceeded last time. The "--forceall" option can be used to enforce the re-run.

Detailed description on output files

5 **1. "final-viral-combined.fa":**

Citation: Jiarong Guo (07/09/2020). Getting started with VirSorter2 . https://dx.doi.org/10.17504/protocols.io.bidpka5n

Identified viral sequences, including two types. Full sequences identified as viral (added with suffix "||full"); partial sequences identified as viral (added with suffix "||{i}index_partial"); here "{i}" can be numbers starting from 0 to max number of viral fragments found in that contig.

Headers of sequences looks likes:

>Caudo-circular||full shape:circular||start:327||end:32076||group:dsDNAphage||score:0.993||hallmark:4

There is a some information in description filed, including: "shape", "start" and "end" position on contig of a viral sequence, best classifier ("group"), "score" from the classfier (ranging from 0 to 1, higher means more like to be viral), number of "hallmark" genes.j

NOTE
Note that classifiers of different viral groups are not exclusive from each other, and may have overlap in their target viral sequence space, which means this info should not be used as reliable classification. We limit the purpose of VirSorter2 to viral idenfication only.

2. "final-viral-score.tsv":

A tab delimited table on score of each viral sequences across groups.

3. "final-viral-boundary.tsv":

Only some of the columns in this file are useful:

- seqname: original sequence name
- trim_orf_index_start, trim_orf_index_end: start and end ORF index on orignal sequence of identified viral sequence
- trim_bp_start, trim_bp_end: start and end position on orignal sequence of identified viral sequence
- trim_pr: score of final trimmed viral sequence
- partial: full sequence as viral or partial sequence as viral; this is defined when a full sequence has score > score cutoff, it is full (0), or else any viral sequence extracted within it is partial (1)
- pr_full: score of the original sequence
- hallmark_cnt: hallmark gene count
- group: the classifier of viral group that gives high score; this should **NOT** be used as reliable classification

NOTE
VirSorter2 tends overestimate the size of viral sequence during provirus extraction procedure in order to achieve better sensitity. We recommend cleaning these provirus predictions to remove potential host genes on the edge of the predicted viral region, e.g. using a tool like CheckV (https://bitbucket.org/berkeleylab/checkv).

Training customized classifier

6 VirSorter2 currently has classifiers of five viral groups (dsDNAphage, NCLDV, RNA, ssNA virus, and*lavidaviridae*). It's designed for easy addition of more classifiers. The information of classifiers are store in the database (-d) specified during "Download database and dependencies" section. For each viral group, it needs four files below:

- model: andom forest classifier model for identifying viral sequences

Citation: Jiarong Guo (07/09/2020). Getting started with VirSorter2 . https://dx.doi.org/10.17504/protocols.io.bidpka5n

- customized.hmm (optional): a collection of viral HMMs for gene annotation; if not specified, the one in "db/hmm/viral/combined.hmm" is used.
- hallmark-gene.list (optional): names of hallmark gene hmm in the above viral hmm database file; These hallmark gene hmms can be collected by literature search or identified by comparing hallmark gene sequences (protein) against HMMs database above withhmmsearch; if not specified, no hallmark genes are counted in feature table
- rbs-prodigal-train.db (optional): prodigal RBS (ribosomal binding site) motif training model; this can be produced with-toption in prodigal; This is useful feature for NCLDV due to large genome size for training; For other viral groups, it's OK to skip this file.

7 Here I will show how to make a model for *autolykiviridae*.

First, prepare the dataset needed: 1) high quality viral genomes 2) protein sequence of hallmark gene; and install two more dependecies.

```
# download genome sequences
wget https://github.com/jiarong/small-dataset/raw/master/autolyki/vibrio_autolyki.fna.gz -O
autolyki.fna.gz

# download hallmark gene seqs
wget https://raw.githubusercontent.com/jiarong/small-dataset/master/autolyki/DJR.fa -O
DJR.fa

# download source code
git clone https://github.com/jiarong/VirSorter2.git # install two more dependencies conda
install -c bioconda -y screed hmmer
```

Then identify hallmark gene HMMs by protein sequences of hallmark genes.

Note that we will need the VirSorter2 database here. If you skip the tutorial above, you can download the database by "virsorter setup -d db -j 4". This will take 10+ mins.

```
# compare all HMMs and protein sequences of hallmark gene# this will take 10+ mins due to
large hmm database file
hmmsearch -T 50 --tblout DJR.hmmtbl --cpu 4 -o /dev/null db/hmm/viral/combined.hmm DJR.fa

# get HMMs names that are signicant hits with protein sequences of hallmark gene python
VirSorter2/virsorter/scripts/prepdb-train-get-seq-from-hmm-domtbl.py 50 DJR.hmmtbl >
hallmark-gene.list
```

With "hallmark-gene.list" and the high quality genomes "autolyki.fna.gz", you can train the features that are used for the classifier model.

```
# train feature file
virsorter train-feature --seqfile autolyki.fna.gz --hallmark hallmark-gene.list --hmm
db/hmm/viral/combined.hmm --frags-per-genome 5 --jobs 4 -w autolyki-feature.out

# check output
ls autolyki-feature.out
```

In the output directory ("autolyki-feature.out"), "all.pdg.ftr" is the feature file needed for next step.

Citation: Jiarong Guo (07/09/2020). Getting started with VirSorter2 . https://dx.doi.org/10.17504/protocols.io.bidpka5n

To make the classifier model, we also need a feature file from cellular organisms. This can be done by collecting genomes from cellular organisms and repeat the above step. Note number of cellular genomes are very large (>200K). Here I will re-use the feature file I have prepared before.

```
# fetch feature file for cellular organisms
wget https://zenodo.org/record/3823805/files/nonviral-common-random-fragments.ftr.gz?
download=1 -O nonviral.ftr.gz
gzip -d nonviral.ftr.gz

# train the classifier model
virsorter train-model --viral-ftrfile autolyki-feature.out/all.pdg.ftr --nonviral-ftrfile
nonviral.ftr --balanced --jobs 4 -w autolyki-model.out
```

In "autolyki-model.out", "feature-importances.tsv" shows the importance of each feature used. "model" is the classifier model we need. Then put the "model" and "hallmark-gene.list" in database directory as the existing viral groups.

```
mkdir db/group/autolykiviridae
cp autolyki-model.out/model db/group/autolykiviridae
cp hallmark-gene.list db/group/autolykiviridae
```

Now you can try this new classifier on the testing dataset, and compare with "dsDNAphage" classifier:

```
# download the testing dataset
wget -O test.fa https://raw.githubusercontent.com/jiarong/VirSorter2/master/test/8seq.fa

# identify viral sequences in testing dataset; it takes 10+ mins;
virsorter run -w autolyki-model-test.out -i test.fa --dbdir db --include-groups
"dsDNAphage,autolykiviridae" -j 4 --min-score 0.8 all

# check the scores in two classifiers
cat autolyki-model-test.out/final-viral-score.tsv
```


Citation: Jiarong Guo (07/09/2020). Getting started with VirSorter2 . https://dx.doi.org/10.17504/protocols.io.bidpka5n