VirusPredictor Manual (Version 1.0)

August 1, 2024

Software introduction

VirusPredictor (version 1.0) is comprised of three sections, including DNA sequence transformation and feature selection; three-class virus prediction XGBoost model training and testing; and six-class subgroup prediction XGBoost model training and testing. VirusPredictor first classifies the query sequences into one of the three classes, i.e., infectious virus, endogenous retrovirus (ERV), and non-ERV human. The predicted infectious virus candidates will then be further classified into one of the six virus taxonomic classes, i.e., dsDNA, ssDNA, Retro-transcribing, ssRNA(-), ssRNA(+), and dsRNA. VirusPredictor is written in Python. Please note this version cannot distinguish other sequences other than these three categories (virus, ERV, and non-ERV human) as every sequence will be classified into one of these three in this version even if it is not any of the three. We are adding an option to mark ambiguous sequence as unknown and adding new functions by expanding the applications to distinguish other sequence categories.

Contributors

Guangchen Liu (gch_liu@163.com)
Xun Chen (xun.chen@uvm.edu)

Citation

Guangchen Liu, Xun Chen, Yihui Luan, and Dawei Li. VirusPredictor: XGBoost-based software to predict virus-related sequences in human data. *Bioinformatics*. 2024. Revision.

Software download

www.dllab.org/software/VirusPredictor.html

Copyright

Update log

Recent major updates

1. Accepted input sequences in both FASTQ and FASTA formats, automatically converted FASTQ to FASTA, and then continued to the next steps.
2. Added input file and output file directories to help users easily add their query sequences and locate the prediction results.
3. Updated software code to automatically identify an input sequence with missing nucleotides and accelerate calculation speed.
4. Corrected several bugs in the Python codes.

Update log

Ver 1.0

08/01/2024 A .tar version is added

02/19/2024 Added three additional test files ("test_virus.fasta", "test_ERV.fasta" and "test_non-ERV.fasta") to the packages.

12/28/2023 Updated model names to be recognized directly by main functions

11/19/2023 Adjusted input-/output-files directory setting to be easily utilized by users

10/28/2023 Updated the training step of both XGBoost models for GPU version users to accelerate model training speed

09/14/2023 Optimized the logic of the Python code

09/14/2023 Optimized the structure of the models

Ver 0.9

06/30/2023 Corrected bugs in the main Python script

05/19/2023 Tested the performance of the six-class model on three gradient length test sequences, i.e., 150-350, 850-950, and 2,000-5,000 bp

04/14/2023 Tested the performance of the three-class model on three gradient length test sequences, i.e., 150-350, 850-950, and 2,000-5,000 bp

Ver 0.8

02/03/2023 Added input file and output file directories to help users easily add their query sequences and locate the prediction results

01/15/2023 Corrected three bugs in the k-tuple method in the Python script

Ver 0.7

11/27/2022 Corrected a bug in the recoding method in the Python script

11/25/2022 Updated the macro average metrics in the Python codes

Ver 0.6

09/20/2022 Utilized random forest algorithm to evaluate the performance of different top features to find the optimal subset of features

08/16/2022 Added ten cut length gradient sequences into the testing datasets

07/11/2022 Added ten cut length gradient sequences into the training datasets

Ver 0.5

05/16/2022 Corrected bugs for checking input files

05/08/2022 Extended the input file format to accept FASTQ format of sequences

05/08/2022 Updated the Python code to automatically identify sequence with missing nucleotides and report a warning in the output file

Ver 0.4

02/23/2022 Added macro average precision, recall, and F1 score metrics for model evaluation

02/17/2022 Utilized MinMaxScaler strategy to normalize the training and testing datasets to improve the models' accuracies

Ver 0.3

12/05/2021 Retrained the models with grid-search strategy to obtain new hyperparameters of the models

11/21/2021 Updated the non-ERV human dataset to train more powerful models

Ver 0.2

09/01/2021 Updated the dimension of input dataset and re-trained the models

08/21/2021 Added three new sequence numerical methods to obtain more information from input sequences for the models

Ver 0.1 released

05/09/2021 Released VirusPredictor Version 0.1 (testing version)

Software environment and installation

OS systems: Linux, Mac, or Windows Version 10 or above. In the following manual, we use Windows Version 10 as an example to demonstrate the workflow.

To use VirusPredictor, Python and the following dependent packages should be installed:

Python: Python Version 3.7/3.8/3.9 (or Anaconda3 2021/2022)
Python packages:

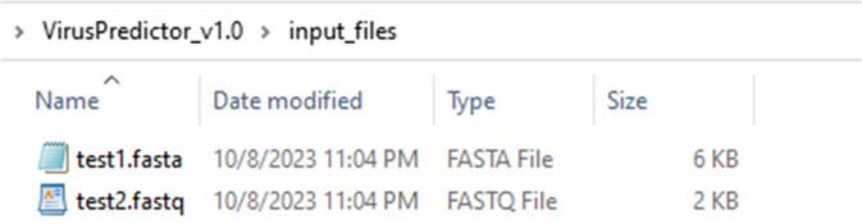
- xgboost version 1.0.2 (*must be this version*)
- scipy version 1.6.2 (*must be this version*)
- biopython version 1.78
- pandas version 0.23.4
- numpy version 1.18.3
- joblib version 1.1.0
- scikit_learn 1.0.2
- openpyxl 3.1.2

Note: Users can use any Python IDE (e.g., Spyder, Pycharm, VScode, and Sublime) to run VirusPredictor. If your versions of Python and/or the above packages are newer than the aforementioned version, please try VirusPredictor first. If you see error messages, please try uninstalling your newer version and instead reinstalling the specified version as described above. For any questions, please contact us for help at gch_liu@163.com.

Software Use

Step 1: Data preparation

Open VirusPredictor_v1.0 package and put your testing sequences into the "*VirusPredictor_v1.0\input_files*" directory (**Figure 1**). VirusPredictor accepts both FASTA and FASTQ format of your testing sequences. If your testing sequences are in FASTQ format, VirusPredictor will automatically transform to FASTA and write to the "*input_files*" directory once you conduct Step2 (i.e., Run work.py).



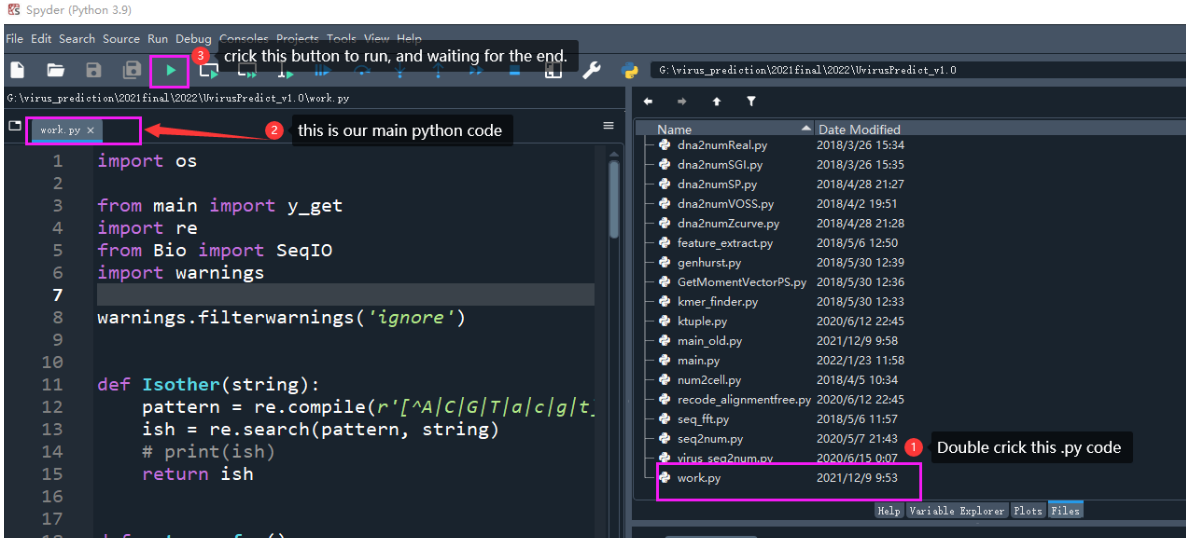
› VirusPredictor_v1.0 › input_files

Name	Date modified	Type	Size
test1.fasta	10/8/2023 11:04 PM	FASTA File	6 KB
test2.fastq	10/8/2023 11:04 PM	FASTQ File	2 KB

Figure 1 Example input files with query sequences.

Step 2: Run work.py

Set directory for Python to make sure your current work directory is in *VirusPredictor_v1.0*. Click **work.py** to run it and wait (**Figure 2**). It will automatically complete the entire prediction procedures. Below is an example using Anacoand3/Spyder interface.


Spyder (Python 3.9)
File Edit Search Source Run Debug Consoles Projects Tools View Help
Help Variable Explorer Plots Files

3 crick this button to run, and waiting for the end.

G:\virus_prediction\2021final\2022\UvirusPredict_v1.0\work.py

work.py

2 this is our main python code

```python
import os

from main import y_get
import re
from Bio import SeqIO
import warnings

warnings.filterwarnings('ignore')


def Isother(string):
    pattern = re.compile(r'[^A|C|G|T|a|c|g|t
    ish = re.search(pattern, string)
    # print(ish)
    return ish
```

G:\virus_prediction\2021final\2022\UvirusPredict_v1.0

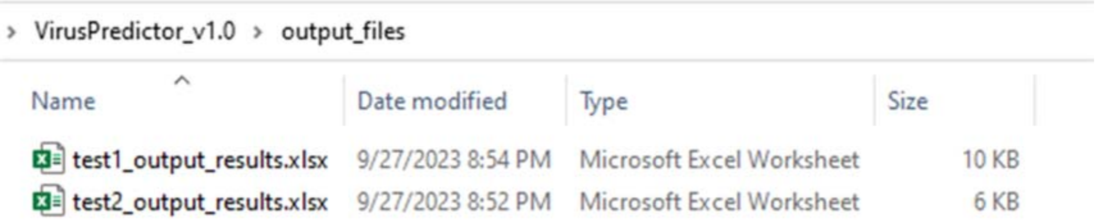
Name	Date Modified
dna2numReal.py	2018/3/26 15:34
dna2numSGI.py	2018/3/26 15:35
dna2numSP.py	2018/4/28 21:27
dna2numVOSS.py	2018/4/2 19:51
dna2numZcurve.py	2018/4/28 21:28
feature_extract.py	2018/5/6 12:50
genhurst.py	2018/5/30 12:39
GetMomentVectorPS.py	2018/5/30 12:36
kmer_finder.py	2018/5/30 12:33
ktuple.py	2020/6/12 22:45
main_old.py	2021/12/9 9:58
main.py	2022/1/23 11:58
num2cell.py	2018/4/5 10:34
recode_alignmentfree.py	2020/6/12 22:45
seq_fft.py	2018/5/6 11:57
seq2num.py	2020/5/7 21:43
virus_seq2num.py	2020/6/15 0:07
work.py	2021/12/9 9:53

1 Double crick this .py code

Figure 2 Example using Anacoand3/Spyder interface.

Step 3: Open results

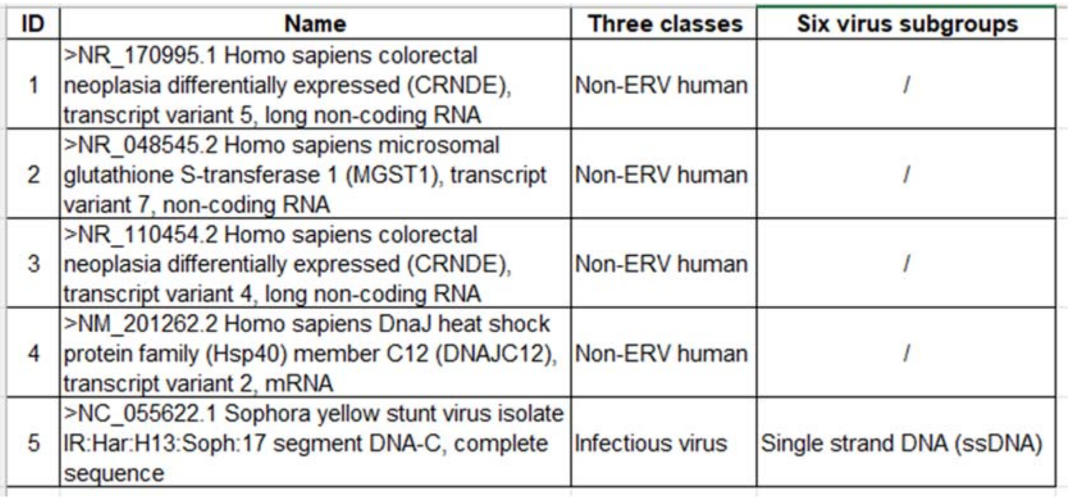
After the work.py ends, the prediction results will be automatically written into the "*VirusPredictor_v1.0\output_files*" directory (**Figures 3** and **4**).



› VirusPredictor_v1.0 › output_files

Name	Date modified	Type	Size
test1_output_results.xlsx	9/27/2023 8:54 PM	Microsoft Excel Worksheet	10 KB
test2_output_results.xlsx	9/27/2023 8:52 PM	Microsoft Excel Worksheet	6 KB

Figure 3 Example output files with prediction results.



ID	Name	Three classes	Six virus subgroups
1	>NR_170995.1 Homo sapiens colorectal neoplasia differentially expressed (CRNDE), transcript variant 5, long non-coding RNA	Non-ERV human	/
2	>NR_048545.2 Homo sapiens microsomal glutathione S-transferase 1 (MGST1), transcript variant 7, non-coding RNA	Non-ERV human	/
3	>NR_110454.2 Homo sapiens colorectal neoplasia differentially expressed (CRNDE), transcript variant 4, long non-coding RNA	Non-ERV human	/
4	>NM_201262.2 Homo sapiens DnaJ heat shock protein family (Hsp40) member C12 (DNAJC12), transcript variant 2, mRNA	Non-ERV human	/
5	>NC_055622.1 Sophora yellow stunt virus isolate IR:Har:H13:Soph:17 segment DNA-C, complete sequence	Infectious virus	Single strand DNA (ssDNA)

Figure 4 Example output file showing predictions. The first two columns show IDs and names of the testing sequences. The prediction results of the three-class XGBoost model are listed in column three. If a sequence is predicted as from a virus, the six-class XGBoost model will be then activated to further predict its viral subgroup as shown in column four.

Frequently Asked Questions

Q1: What is the requirement of query sequences? How do you deal with missing nucleotides in query sequences?

Answer: The query sequences must be in FASTA or FASTQ format, and must be continuous and cannot contain carriage returns, spaces or any other symbols except nucleotides A/C/G/T/a/c/g/t. VirusPredictor automatically converts a/c/g/t to A/C/G/T. If a sequence contains missing nucleotides N, or any other symbols except nucleotides A/C/G/T/a/c/g/t, VirusPredictor will discard this sequence and generate a warning message in the output file. In case there are “U”s in your RNA sequences, please convert “U” to “T” first.

Q2: Where can I find human ERV, and non-ERV human sequences?

Answer: Our ERV sequences (hg38_ERV_100bp.fa; min length = 100 bp) and non-ERV human sequences (hg38_rmsk_ERV.fa; min length = 100 bp) in FASTA can be downloaded from our website (www.dllab.org/software/VirusPredictor.html).

Q3: Which sequence lengths are most useful for accurate prediction purposes?

Answer: Since the prediction accuracies increase as the sequences become longer, we suggest assembling Illumina short reads into contigs, e.g., ~1,000 bp (at least ~800 bp) or longer sequences, whenever possible before predictions. The longer the input sequences, the higher the prediction accuracies.

Q4: How to correctly interpret the results from VirusPredictor?

Answer:
ERVs in the human genome: The human reference genome contains both ERV and non-ERV sequences. Because ERVs are under-studied, many ERVs have not been identified or annotated in the human genome. Because some “non-ERV” sequences from the ERV-masked human genome are still indeed ERV sequences, we anticipate that some of these “non-ERV” sequences are predicted as ERVs when the query sequences are short.

Average accuracies: The accuracies presented in our paper were the average values of a large number of testing sequences. Thus, a small number of testing sequences may result in deviations from our reported average accuracies.

Q5: Can I use VirusPredictor to train my own classification models for other species, such as classifications of virus vs. bacteria, and of dsDNA subfamilies?

Answer: Sure, VirusPredictor supports users to customize and use their own datasets (as long as each sequence has a label) to train new models for their own classification projects. To do this, just prepare your own data (e.g., multiple files separately) in FASTA format and put them under the main directory. Run *train.py* and you will obtain two model related results, such as “*model_xgboost_3labels_vNCBI_nonLTR_LTR_model_4_GPU.m*” and “*scaler_3labels_vNCBI_nonLTR_LTR_model_4_GPU.m*”. Put these two results into the “*\VirusPredictor_v1.0\model*” directory, then you run *work.py* to test your model described in this manual. VirusPredictor can be expanded to other sequence classification studies such as classification of virus vs. bacteria using metagenomic data. Our VirusPredictor open-source pipeline has built-in DNA sequence transformation and feature selection. Users can also replace XGBoost with other machine learning algorithms to build customized models.

Q6: How can I use GPU to train the models?

Answer: To accelerate model training speed, GPU users may use the code "*model = xgb.XGBClassifier(tree_method='gpu_hist')*" (see train.py line 164) in the training step for both XGBoost models.

Q7: If I have other questions about using this software, how could I get help?

Answer: Please contact with Dr. Liu: gch_liu@163.com or Dr. Li: dawei.li@ttuhsc.edu for any bugs, questions, or suggestions, and we will help you as soon as possible.